EXPLOITING CROSS-TECHNOLOGY INTERFERENCE FOR EFFICIENT NETWORK SERVICES IN WIRELESS SYSTEMS

By

Ruogu Zhou

A DISSERTATION

Submitted
to Michigan State University
in partial fulfillment of the requirements
for the degree of

Computer Science - Doctor of Philosophy

2014

ABSTRACT

EXPLOITING CROSS-TECHNOLOGY INTERFERENCE FOR EFFICIENT NETWORK SERVICES IN WIRELESS SYSTEMS

By

Ruogu Zhou

In the last decade, we have witnessed the wide adoption of a variety of wireless technologies like WiFi, Cellular, Bluetooth, ZigBee, and Near-field Communication(NFC). However, the fast growth of wireless networks generates significant cross-technology interference, which leads to network performance degradation and potential security breach. In this dissertation, we propose two novel physical layer techniques to deal with the interference, and improve the performance and security of sensor networks and mobile systems, respectively. First, we exploit the WiFi interference as a “blessing" in the design of sensor networks and develop novel WiFi interference detection techniques for ZigBee sensors. Second, utilizing these techniques, we design three efficient network services: WiFi discovery which detects the existence of nearby WiFi networks using ZigBee sensors, WiFi performance monitoring which measures and tracks performance of WiFi networks using a ZigBee sensor network, and time synchronization which provides synchronized clocks for sensor networks based on WiFi signals. Third, we design a novel, noninvasive NFC security system called *nShield* to reduce the transmission power of NFC radios, which protects NFC against passive eavesdropping. nShield implements a novel adaptive RF attenuation scheme, in which the extra RF energy of NFC transmissions is determined and absorbed by nShield. At the same time, nShield scavenges the extra RF energy to sustain the perpetual operation. Together with the extremely lo-power design, it enables nShield to provide the host uninterrupted protection against malicious eavesdropping. The above systems are implemented and extensively evaluated on a testbed of sensor networks and smartphones.

This dissertation is dedicated to my family and friends.

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my gratitude to everyone who helped me during the past five years.

My deepest gratitude goes to my advisor Dr. Guoliang Xing, for his patience, support, trust and friendship, which has been extremely valuable for my academic study as well as my personal life. I am grateful for the freedom he offered, with which I can pursue various projects that I like to work on without objection, not to mention his insightful advice that helped me overcome numerous problems during these projects. I feel extremely fortunate to have him as my advisor, and hope someday I could become a guy like him.

I am also grateful to my Phd Committee members, Dr. Eric Torng, Dr. Tongtong Li, and Dr. Li Xiao, for their insightful criticism and inspiring advice. Dr. Eric Torng is probably the best teacher I had during my PhD study. He sets high standards in his course for his students, but he does whatever he could do to help them get over these standards. He offered invaluable advice on Chapter 3 of this dissertation. Dr. Tongtong Li provided many practical advices not only to my dissertation, but also on how to conduct research during the early stage of my PhD study. Her lecture is also top-notch, from which I learned a lot(and applied in my research). Dr. Li Xiao, unlike most other professors who are usually commanding, is a very gentle professor. I always feel warm and relax when talking to her. I am indebted for her pointed comments and kind encouragement to my dissertation and research.

I also want express gratitude to my current and former colleagues at ELANS Lab, including but not exclusive to Rui Tan, Jun Huang, Pei Huang, Jinzhu Chen, Tian Hao, and Yu Wang, for their support and help. From the collaboration with them, I learned a lot on systematical methods and skills to solve hard problems. It has been really fun and rewarding to work with them. Kudos to Pei Huang and Jinzhu Chen, who have recently earned their doctoral titles!

I want to say thank you to all my friends in United States and China(a long list, but you know

you are here), for their support and the belief in me. They helped me a lot on maintaining my sanity and keeping myself on path during those hard times. I cannot imagine what I would become without those parties, games, and excursions that we had together, and I will forever remember those great moments. The friendship with them is one of the best parts in the five past years.

Last but not the least, none of these would be possible without the love and support of my family. Both my immediate family and my extended family gave me unwavering and unconditional love, support and patience, no matter how I did during the past five years. I especially want to thank my best friend, soul-mate, and wife, Jing Yang. The five past years have proved to be quite a rough ride and we had both good times and bad times together. However, she always has faith on me, even when I feel lost and don't have faith on myself. I am very grateful for Jing to be sticking closely with me during this long trek, and provide her unwavering love which is all that matters to me. Another special thank is for my dear daughter Eva, who always cheers me up with her sweet smiles during my dark times.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Motivation

In the last decade, the advancement of technologies has significantly bridged the performance gap between wireless technologies and wired communication approaches in terms of bandwidth, delay, security, and etc. As a result, wireless technologies like WiFi, Cellular, Bluetooth, ZigBee, and NFC (Near-field communication) are being widely adopted worldwide to replace and complement existing wired communication infrastructures. For example, the number of WiFi hotspots is expected to grow from 1.3 in 2011 to 5.8 million in 2015 [31]. However, a key challenge in the design of wireless systems is the interference that usually causes communication performance degradation. In particular, in public areas such as offices, college campuses, and hospitals, the WiFi networks that provide Internet access for mobile devices are often co-located with sensors embedded in the environment, such as those offering security surveillance [54] and urban environmental monitoring [91] services. When sensor networks work in overlapping or adjacent frequency bands with WiFi networks, their performance could be significantly impaired by the WiFi interference [55]. Besides performance degradation, interference could also lead to security and privacy issues [97]. An example is the short range NFC technology that has been widely adopted by security sensitive applications like contactless payment. The interference generated by NFC usually carries security-sensitive information, e.g., identity, credit card information, etc. As wireless interference can be easily detected by unintended receivers (i.e., eavesdropper), such sensitive data may be leaked to malicious attackers.

Many existing approaches [49][92][48][56] strive to mitigate the performance loss on wireless sensor networks caused by WiFi interference. Most of them consider the interference as “curse" to wireless networks as they harm the system performance. We argue that there exist abundant

opportunities for sensor networks to exploit such interference as a “blessing". Sensors are usually capable of simple spectrum sensing. Using the received signal strength indication (RSSI) register which is provided by most commercial off-the-shelf ZigBee radios, ZigBee radios can sense the WiFi interference signals and hence are able to retrieve many useful features from the interference. For example, the periodic beacon frames transmitted by WiFi Access Points (APs) can cause periodic interference patterns in the RSS series received by ZigBee radios, which can be used, for example, as time synchronization reference clock on sensor network platforms. We propose to exploit such cross-technology interference between WiFi networks and sensor networks to design several important network services. These services include:

1. **W**iFi discovery. Due to the limited coverage of existing WiFis, WiFi-enabled devices (e.g., laptops and smartphones) must actively discover new WiFi access points (APs) once they leave the coverage of current network. Such an approach wastes the precious energy of mobile devices due to excessive listening and scanning operations of WiFi network interface cards (NICs). Several solutions have been proposed to address this issue. The first approach utilizes a secondary low-power radio that communicates with peer radios on WiFi APs to find connectivity opportunities or reduce the energy consumption of data transfers [83][84][34][33] [72][69][41]. However, this approach requires significant modifications to existing network infrastructures. The second approach predicts the availability of WiFi based on *context* information. Cellular cell-tower information [35] or together with Bluetooth contact-patterns [75] have been used to improve WiFi prediction accuracy. However, such a context-aware approach requires extensive training based on historical information and hence is not feasible in unknown environments.

2. **W**iFi network performance monitoring. Compared with wired LANs, WiFi suffers significantly higher level of *spatial* and *temporal* performance variability. Due to the broadcast nature of wireless channel, signal propagation are susceptible to environmental conditions. As a result, end-users often experience highly variable signal quality. To diagnose such

transient service degradation and plan for future network upgrades, it is essential to closely monitor the spatial and temporal performance of a WiFi network as well as to collect the statistics of its users. The existing WiFi performance monitoring solutions [39][40][43] rely on 802.11-based listening devices. However, due to the high power consumption of 802.11 radios, the monitoring nodes must be connected to external power supplies (e.g., wall power or desktop computers). This constraint leads to high installation costs and poor spatiotemporal monitoring granularity.

3. **T**ime synchronization for sensor networks. Time synchronization is a fundamental service for Wireless Sensor Networks (WSNs). A number of protocols are available to synchronize clocks of nodes through message passing, which include RBS [45], TPSN [47] and FTSP [67]. However, they incur high messaging overhead or poor accuracy in large-scale WSNs. For instance, an in-depth analysis showed that the clock skew in FTSP grows exponentially with network diameter [**?**]. An alternative approach is to leverage external global timebases such as those extracted from Global Positioning System (GPS), timekeeping radio stations (e.g., WWVB in the US and JJY in Japan), FM Radio Data System [63], or even power grid [79]. This approach largely reduces the overhead of message exchanges. However, they require hardware receiver to decode the out of band clock signal, introducing extra cost and design complexity. Moreover, the signals of GPS and WWVB have poor penetration through walls while the electromagnetic field is strong enough for clock calibration only in the vicinity of power lines [79].

Previous work [53] aims to preserve the data security of the NFC interference signals by enhancing NFC data encryption. However, without reducing the amount of interference signals generated by NFC radios, they cannot completely prevent information leakage. Moreover, these security protections are not properly implemented [53][50] on many systems and are hence susceptible to security attacks. Our measurement study on the NFC physical security reveals that the power of NFC interference signals can be significantly reduced by using physical protection approaches like

signal attenuation. We propose to utilize signal attenuation to preserve the data security of NFC systems. Combined with proper encryption, such a physical protection approach will drastically improve the security of NFC systems.

1.2 Contributions

In this dissertation, we propose to exploit WiFi interference signals in the design of sensor network services and to develop a security system that reduces signal power of NFC interference and prevents malicious eavesdropping. The contributions of this dissertation include:

1. We develop novel WiFi interference detection techniques for ZigBee sensors to identify WiFi interference. We develop a new digital signal processing algorithm called Common Multiple Folding (CMF) that accurately amplifies periodic beacons in WiFi interference signals. We also adopt a constant false alarm rate (CFAR) detector that can minimize the false negative (FN) rate of WiFi beacon detection while satisfying the user-specified upper bound on false positive (FP) rate.

2. We develop a system called *ZiFi* that identifies the existence of WiFi networks using the WiFi interference detection technique described above. We have implemented ZiFi on two platforms, a Linux netbook integrating a TelosB mote through the USB interface, and a Nokia N73 smartphone integrating a ZigBee card through the miniSD interface. Our experiments show that, under typical settings, ZiFi can detect WiFi APs with high accuracy ($< 5\%$ total FP and FN rate), short delay (~ 780 *ms*), and little computation overhead.

3. We develop a new ZigBee-based WiFi performance monitoring system called WizNet. Powered by batteries, ZigBee sensors of WizNet can be deployed in large quantities to monitor the spatial performance of a WiFi in long periods of time. We adopt the CMF algorithm and a novel RSS sequence alignment algorithm for automatic WiFi AP identification and tracking. To ensure the monitoring fidelity, we exploit the frequency diversity and spatial diversity of the WiFi interference signals to account for the significant differences in ZigBee and WiFi

radios, such as bandwidth and susceptibility to multipath and frequency-selective fading. We also derive a simple yet accurate linear estimator from a signal propagation model for inferring the access points' signal to noise ratio (SNR). Moreover, we develop a technique for collecting WiFi client statistics and classifying device models based on RSS signatures of 802.11 access point scans. We have implemented WizNet in TinyOS 2.x and extensively evaluated its performance on a wireless testbed. Our results over a period of 140 hours show that WizNet can accurately capture the spatial and temporal performance variability of a large-scale production WiFi.

4. We experimentally characterize the spatial and temporal characteristics of WiFi beacons in an enterprise WiFi network consisting of over 50 APs deployed in a 300,000 square foot office building. We implement a time synchronization protocol called WizSync in TinyOS 2.1x, which exploits periodic WiFi beacons as synchronization reference clock. We conduct extensive evaluation on a testbed consisting of 19 TelosB motes. Our results show that WizSync can achieve an average synchronization error of 0.12 milliseconds over a period of 10 days with radio power consumption of 50.9 microwatts/node.

5. We propose a novel, noninvasive NFC security system called *nShield* to reduce the amount of interference signals generated by NFC radios, which protects NFC against passive eavesdropping. nShield is a credit card-sized thin pad that can be easily stuck on the back of mobile devices (see Fig. 7.6). nShield implements a novel adaptive RF attenuation scheme, in which the extra RF energy of NFC transmissions is determined and absorbed by nShield. At the same time, nShield scavenges the extra RF energy to sustain the perpetual operation. A key contribution of this work is the analysis of the factors affecting the energy harvesting efficiency, and the design of a highly effective energy harvesting system. nSheild is capable of harvesting significant amount of power (55 mW) from commodity mobile devices, which is at least a 1.8X improvement over the state-of-the-art NFC-based energy harvesting systems. Together with the extremely lo-power design, it enables nShield to provide

the host uninterrupted protection against malicious eavesdropping. Lastly, the small form factor, self-sustainability, and transparency to OS, makes nShield an attractive solution to retrofit existing mobile devices with protection against passive eavesdropping.

CHAPTER 2

RELATED WORK

In the following, we discuss the work related to each of the systems proposed in this dissertation, respectively.

2.1 WiFi Network Discovery

The idea of waking up high-power radio using a secondary low-power radio was first proposed in Wake-on-Wireless [83]. Several recent systems including On-Demand-Paging [34], Cell2notify [33], and CoolSpots [72], also propose to use a secondary radio to either help detect WiFi signal or reduce the energy consumption of WiFi data transfers. Wake-on-WLAN [69], S-WOW [70], and Esense [41] allow ZigBee and WiFi radios to communicate through sensing specially designed codes. However, the above solutions suffer from at least one of the following issues. (1) They assume a “cooperative" setting where substantial software and/or hardware modifications to existing network infrastructures can be made, which hinders their wide deployment. (2) As the secondary low-power radio has significantly shorter communication range, the existing solutions often rely on additional proxy servers to achieve satisfactory network discovery range. In contrast to these solutions, ZiFi completely relies on the ZigBee interface on WiFi clients to detect the existence of WiFi APs and requires no modification to WLAN infrastructure. Moreover, ZiFi detects WiFi signal by passively sensing its energy, which ensures a similar detection range as WiFi interface.

There exist portable spectrum scanners [10] [14], often referred to as *WiFi detectors*, which are specially designed to find WiFi signal. They usually work in standalone mode but may also be modified to wake up WiFi NICs on mobile devices [84]. However, they require the use of 802.11 radios and hence come with high power consumption. Moreover, as specialized hardware, they have not gained popularity in WiFi community.

In [71], BreadCrumbs is proposed to build the mobility model of a mobile device by tracking

its movement from GPS and use the model to predict the connectivity opportunities. Cellular cell-tower information or together with Bluetooth contact-patterns have been used in [35][75] to predict the existence of WiFi. A key drawback of these context-aware approaches is that they rely on historical information and hence cannot be applied to unknown environments. Moreover, they often require extensive offline training in order to achieve satisfactory runtime prediction accuracy. In contrast, ZiFi detects WiFi coverage by in-situ processing of signals transmitted by APs without offline training or context information collection.

2.2 WiFi Network Performance Monitoring

Performance measurement and monitoring are critical for WiFi infrastructure. The existing solutions can be classified into three basic categories.

The first approach consists of various WiFi site-survey tools including Fluke Airmagnet [9], Berkeley Varitronics Swarm [5], and Airtight Networks[3]. These commercial products are typically expensive. For example, a single Fluke Airmagnet Express field kit costs over $5,000 [9]. Moreover, they need to be carried by experienced engineers who roam about the site to measure the network performance. Several recent efforts studied urban-scale WiFi coverage in war-driving experiments [77][37]. The above site-surveying approaches incur high labor costs and hence are not suitable for long-term and real-time WiFi performance monitoring. In the Sybot system [59], mobile robots carrying 802.11 radios can assess the WiFi networks in a building. Because of the intrusive nature and the challenge of motion planning in complex environments, the use of survey robots may not be feasible for large-scale enterprise WiFi deployments.

The second approach exploits the already available network infrastructure or installs dedicated 802.11 nodes for distributed WiFi performance monitoring. The DAIR system [39] takes advantage of networked desktop computers equipped with WiFi network measurement devices for long-term WiFi monitoring. Although these systems can assess the spatial performance of a network in real-time, their spatial granularity is constrained to the locations where 802.11 computers are available. Moreover, installing monitoring devices or software brings privacy concerns and may make desk-

top users reluctant to participate. Several other systems [95][66][43] deployed dedicated 802.11 nodes for indoor spatial network performance monitoring. This approach is also employed by the Argos system[78] to monitor urban-scale WiFi networks in outdoor environment. However, due to the high power consumption of 802.11 NICs, the monitoring nodes must be plugged to wall power, which not only limits the coverage but also incurs high installation costs.

The third approach utilizes non-802.11 nodes for WiFi monitoring. Similar to this work, WiBee [90] adopts ZigBee nodes to build real-time WiFi radio RSS maps. However, WiBee does not consider the significant bandwidth difference of heterogeneous radios as well as the indoor frequency-selective fading. As a result, it suffers from large estimation errors (as high as 15 dB) which limits its practical use. Moreover, WiBee only focuses on building coarse-grained RSS map of WiFi while WizNet can monitor fine-grained performance characteristics including SNR, channel utilization rate, and client statistics.

2.3 Time Synchronization for Sensor Networks

Time synchronization in distributed systems has been extensively studied. The existing approaches fall into two broad categories. The first category includes the approaches based on network-wide message exchanging. In the second category, time synchronization is achieved with the assistance of more accurate external clock signals.

Several clock synchronization protocols based on message passing have been developed for WSNs. They leverage message passing between nodes to measure and eliminate the time jitter of various sources. In RBS [45], receivers correct their pairwise clock skew by exchanging the receiving times of the broadcast reference message. Different from RBS, TPSN [47] and FTSP [67] employ timestamping of message exchanges between a sender and receiver to eliminate sources of time jitter. A key drawback of these message passing protocols is that they incur prohibitively high overhead or poor synchronization accuracy when the network size scales to more than tens of nodes. For instance, an in-depth analysis showed that the clock skew in FTSP grows exponentially with network diameter [**?**].

Another time synchronization approach takes advantage of the global timebases induced by various infrastructures including Global Positioning System (GPS), timekeeping radio stations (e.g., WWVB in the US and JJY in Japan),FM Radio Data System [63] or even power grid [79]. These solutions largely reduce the message exchanges in the network by synchronizing the nodes to a global time reference. However, they require extra hardware receivers to decode the global clock signal. GPS and WWVB signals contain highly accurate global time reference. However, they cannot penetrate building walls well and hence are largely limited to outdoor applications. For instance, a recent empirical study in indoor environments [42] shows that the WWVB signal is only received in 47% of the time. Several recent time synchronization systems exploited the infrastructure of buildings. In RT-Link [80], the wiring infrastructure of building is used as antenna of AM radio to broadcast global time beacons for synchronizing the clocks of sensor nodes. The Syntonistor system synchronizes nodes' clocks to periodic signals extracted from electromagnetic field of AC powerlines [79]. However, due to the significant attenuation of electromagnetic field, Syntonistor can only work effectively in the vicinity of power lines. Moreover, the performance of the system may suffer from interference of other electromagnetic sources.

Recent work exploits the use of two clocks for time synchronization [60, 82]. In the HARMONIA system [60], the high-frequency microcontroller clock synchronizes an accurate but low-frequency (1 Hz) Real Time Clock (RTC). In [82], a fast clock (e.g., 8 MHz FPGA-based clock) is periodically activated to assist a slow clock (e.g., 32 KHz crystal oscillator) to achieve high-resolution radio timestamping. Without a global time reference, these systems still require message passing among nodes to calibrate their clocks.

2.4 NFC Security

Near Field Communication (NFC) is a new short-range wireless communication standard evolved from HF RFID technology. Several studies have been conducted on the distance of eavesdropping RFID proximity cards. In [51], the authors measure the passive eavesdropping distance of the communication between a commercial reader and a Philips Mifare card using a wide band sniffer.

		nShield	EnGarde
NFC radio	Radio type	Software-define radio	Dedicated ASIC NFC radio
	TX capability	Supports NFC-A (implemented), NFC-B, NFC-F HW accelerated SW encoding	Jamming only No TX support
	RX capability	Supports NFC-A (implemented), NFC-B, NFC-F HW accelerated SW decoding	NFC-A, NFC-B, and NFC-F HW decoding
Energy harvesting	Ant. configuration	Dual antenna	Dual antenna
	Optimization	High Q antenna Voltage matching	N/A
	Harvestable power	55 mW constant	maximum 30 mW transferred to antenna
	Max initiator duty-cycle	100% (tag-emulation)	66% (subcarrier)
System pwr consumption	Active	8.7 mW	32.7 mW
	Sleep	23 uW	38.8 uW

Table 2.1: Comparison of hardware of nShield and EnGarde.

The results show that the possible eavesdropping distance is more than 4 m [51]. In [53], the authors analyze the security of NFC and estimate the passive eavesdropping distance of NFC to be about 10 m. However, this result is not experimentally validated. In [61], the maximum passive eavesdropping distance of NFC is empirically measured to be 30 cm using Mifare tags and an oscilloscope. However, the antennas of Mifare tags used in their experiments are not optimized for eavesdropping. To our best knowledge, our work is the first empirical study on the practical passive NFC eavesdropping distance under realistic experimental settings. We have designed and implemented a prototype NFC sniffer. Its small form factor and high sensitivity demonstrated the feasibility of launching passive eavesdropping attack from distance. In particular, we are able to achieve a 2.4 m eavesdropping distance with our portable NFC sniffer (see Section 7.3).

Several approaches have been proposed to protect NFC from malicious attacks. A common solution is to modify the OS of mobile devices [53] to enhance the security of NFC. However, the mobile device would become vulnerable if the integrity of the OS is compromised (e.g., by rooting the device)[50]. To address this issue, several systems adopt additional hardware security devices. RFID guardian [76] provides protection by actively jamming suspicious NFC transactions.

However, active jamming consumes considerable power and requires bulky hardware (e.g., RF amplifier and large battery), which significantly limits RFID guardian's applications. Proxmark III [22] is a widely used RFID/NFC software defined radio that is capable of detecting an attack, and generating jam signals. However, it must be plugged in as its FPGA-based design consumes significant power (about several hundred milliwatts). Furthermore, none of these approaches can provide anti-eavesdropping protection.

NFC is ideal for energy harvesting, due to the condensed RF field strength generated by its high transmission power and short communication range. Energy harvesting enables a mobile device to replenish its energy in the presence of NFC RF field. The NFC Discover kit [26] from ST include a sensor board can be wirelessly powered by nearby NFC initiators. NFC-WISP [44][19] is a software defined passive tag platform, which is capable of harvesting energy from NFC transmissions and conducting simple sensing and computational tasks. A key difference between the energy harvesting component of nShield and the above two systems is the amount of power harvested. With extensive optimizations to harvesting antenna and energy management circuit, nShield can harvest a power of about 55 mW, compared to mere 10.2 mW and 17.7 mW of NFC Discover kit and NFC-WISP, respectively. The significant improvement on the energy harvesting efficiency enables nShield to power additional components and perform sophisticated operations to ensure system security.

To date the most relevant work to ours is EnGarde [50]. EnGarde is a hardware NFC security device that jams ongoing malicious NFC transactions. Different from RFID guardian and Proxmark III, EnGarde is optimized for mobile devices and harvests energy from NFC transmissions. However, EnGarde protects NFC by censoring the content of NFC transactions, and hence cannot defend against eavesdropping attacks. We provide a comparison between the hardware of the two systems, which is summarized in Table 2.1.

nShield is built based on a software-define radio (SDR), which is capable of transmitting to and receiving from NFC initiators. The SDR can be programmed to support standard and custom protocols. However, as SDR relies on software radio stack to decode and encode messages, it tend-

s to incur longer delays. In the case of nShield, hardware components (demodulator, modulator, etc.) are utilized to accelerate the encoding/decoding, which significantly reduces the delay. EnGarde, on the other hand, employs a hardware-based NFC transceiver (TI TRF7970A) that incurs shorter delay than SDR-based transceiver. However, EnGarde only employs the receiving chain of the hardware transceiver, due to its dual antenna configuration. Although EnGarde implements a simple transmitter that can generate jamming signals, it does not support data transmissions. The capability of transmission is critical for tag emulation, which increases the amount of energy harvested from initiator significantly. Another disadvantage of this configuration is the resulted high power consumption, since the hardware transceiver employed by EnGarde is mainly designed for power-hungry NFC initiators. Moreover, the hardware-based transceiver does not support the development of new physical and link-level protocols.

The energy harvesting system of nShield also differs significantly from that of EnGarde. Although a dual antenna configuration is employed by both systems, it is used to meet fundamentally different requirements. Specifically, EnGarde employs the dual antenna configuration for tag proximity detection, while nShield adopts it for improving power harvesting efficiency. The harvesting antenna of nShield is specially designed to achieve high Q-factor. nShield also employs a technique called voltage matching, which carefully matches the output voltage of the antenna to that of the battery to maximize the amount of power harvested. On the another hand, EnGardes does not perform any load-source matching, which significantly limits the power harvesting efficiency. Moreover, EnGarde does not support tag emulation due to the lack of transmission capability, and can only trigger the initiator to raise its duty-cycle to 66% by using jamming. This further lowers the amount of energy harvested. Lastly, the active power consumption of EnGarde is much higher than nShield (32.7 mW vs 8.7 mW), due to the use of hardware-based transceiver.

CHAPTER 3

DETECTING WIFI INTERFERENCE WITH ZIGBEE SENSORS

Due to the unlicensed 2.4G spectrum, the interference received by ZigBee radios may contain signals from various devices, such as WiFi APs, Bluetooth headsets, and etc. As a result, in order to utilize WiFi interference in sensor networks, ZigBee sensors must be capable of identifying the WiFi interference signals. However this is challenging since ZigBee radios cannot directly decode WiFi frames, and RSS statistics (power magnitude, time duration, inter-arrival gap, and etc.) provide little information about the nature and source of incoming signals. Therefore, the first task of this dissertation is to investigate an approach for WiFi interference detection using ZigBee sensors, which we discuss in this chapter.

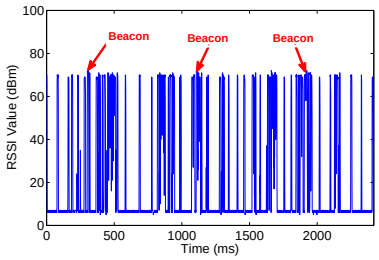
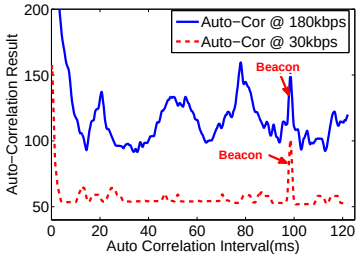
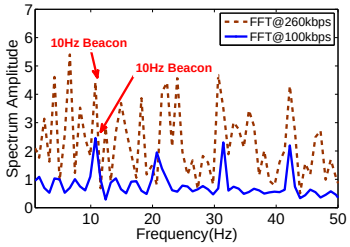


Figure 3.1: The RSS samples collected by a CC2420 radio when a nearby 802.11 AP transmits.



(a) Autocorrelation, 30 and 180 Kbps.



(b) FFT, 100 and 260 Kbps

Figure 3.2: Performance of Autocorrelation and FFT under different traffic rates.

When 802.15.4 radio operates on the same or adjacent channels as 802.11 APs, the signals from 802.11 APs can be sensed through RSSI. Fig. 4.3 shows the time series of RSS samples gathered from a TelosB mote equipped with ZigBee-compliant CC2420 radio when a nearby 802.11 AP actively transmits. However, as ZigBee radios cannot directly decode 802.11 frames, RSS statistics provide little information about the nature and source of incoming signals. We address this challenge by searching for unique interference signatures in RSS samples.

802.11 beacon signals can cause unique interference patterns in RSS samples, which can be detected by ZigBee radios. Several properties of 802.11 beacons make them ideal for detection.

First, they are broadcast *periodically* by APs and hence lead to periodic traces in RSS samples. Second, beacons are typically broadcast at the *lowest* data rate, which makes it easier to capture by the RSSI register of low-rate ZigBee radios. However, a key challenge is that, thousands of data frames are typically transmitted between two beacon frames, causing heavy noise in RSS samples. Moreover, the RSS time duration provides little hint as there is a large overlap between the in-air times of data and beacon frames. Several signal processing techniques such as *Fast Fourier Transformation (FFT)* and *Autocorrelation*, can be used to detect periodic patterns from noisy measurement. However, our experimental results in Section 4.3 show that their performance are highly sensitive to the intensity of noise, making them ill-suited for identifying beacons in moderate to high traffic workload. Moreover, both of them impose high computation overhead for mobile devices like smartphones. We adopt a novel digital signal processing algorithm called Common Multiple Folding (CMF) that can reliably identify periodic WiFi beacons at small delay and overhead. Fig. 3.2 shows the performance of autocorrelation and FFT under different noise levels. As can be seen, the results of autocorrelation and FFT can clearly reveal the beacons when the data rates are 30 Kbps and 100 Kbps, respectively. However when the data rates increase to 180 Kbps and 260 Kbps, respectively, both of them fail to detect beacons. In the following, we discuss the details of our WiFi interference identification algorithm.

3.1 RSS Sampling and Shaping

The RSS sampler of ZiFi reads the RSSI register of ZigBee radio every T *us* for total D *us*. T and D are referred to as RSS *sampling period* and *sampling window size*, respectively. The sampling period should be short enough to capture the transmission of 802.11 beacon frames. Each sample contains the RSS value averaged for a number of incoming symbols. For instance, the built-in RSS register in the CC2420 radio is updated every 32 *us* when enabled and the value is averaged for 8 symbols (128 *us*). However, reading the RSS register every 32 *us* (i.e., $T = 32$) incurs unnecessarily high computation and memory overhead. Intuitively, the sampling period should be short enough to capture the transmission of 802.11 beacon frames. We now discuss how to

determine the sampling period based on beacon frame length and channel transmission rate. The 802.11 beacon frame contains a preamble, a physical layer convergence procedure (PLCP) header, and a MAC frame. The preamble and PLCP header are broadcast at a fixed rate of 1 Mbps while the transmission rate of MAC fame varies with the version of protocol. The in-air time of a beacon frame (denoted by T_{beacon}) can be calculated as:

$$T_{beacon} = \frac{L_{preamble} + L_{PLCP}}{B_p} + \frac{L_{MAC}}{B_m} \tag{3.1}$$

where $L_{preamble}$, L_{PLCP}, and L_{MAC} represent the numbers of bits in preamble, PLCP header, and MAC frame, respectively. B_p and B_m represent the transmission rates of preamble/PLCP header and MAC frame, respectively. The PLCP header has 48 bits while the formats of preamble differ in different 802.11 protocols. We only consider two 802.11b and 802.11g-compatible preamble formats, specifically long preamble(144 bits) and short preamble (72 bits), as they are widely adopted in practice for maximum network compatibility. A MAC frame should have at least 120 bytes, to account for the necessary parameters included in a beacon including SSID, supported rates etc. Several data transmission rates are available in 802.11 protocol family. However, due to the importance of beacon frames, they are always broadcast at the lowest possible rate that is supported by the protocol, typically 1 Mbps in 802.11b and 6 Mbps in 802.11g. As discussed above, the minimum in-air time of a beacon frame can be calculated as 256 *us* when $L_{preamble} = 72$ bits, $L_{PLCP} = 48$ bits, $L_{MAC} = 120$ bytes, $B_p = 1$ Mbps and $B_m = 6$ Mbps. We choose a sampling period of 122 *us* on the TelosB platform, which is 4 ticks of the on-board clock of TelosB. As a result, for every beacon frame, we will at least capture two RSS samples.

After enough RSS samples are collected, the RSS shaper adjusts the power magnitude of them to mitigate the noise in the following beacon detection stage. The shaper applies the following two criteria to RSS samples in order: 1) The magnitude of an RSS sample is set to zero if it is below -90 dBm because, even if the sample contains beacon, a low RSS indicates poor signal quality from the AP and low probability of successful client association. 2) The magnitude of all remaining RSS samples are set to 1 mW. 3) The magnitude of S consecutive non-zero samples will be set

to zero if $S \notin [s_1, s_2]$. A cluster of such samples is typically generated by WiFi data traffic. We now discuss how to determine s_1 and s_2 based on beacon size and 802.11 transmission rates. The 802.11 beacon frame has a size between 80 and 200 bytes. When possible 802.11 transmission rates are considered, the in-air time of a beacon frame is from 256 to 1720 *us*, which corresponds to an RSS sample count in $[\frac{256}{T}, \frac{1720}{T}]$, where T is the RSS sampling period. Therefore, a number of consecutive samples can be removed if the count lies outside this range. After the above three steps, the magnitude of RSS samples is either 0 or 1 and the number of consecutive non-zero RSS samples is within $[\frac{256}{T}, \frac{1720}{T}]$.

3.2 Common Multiple Folding Algorithm

We have developed a novel digital signal processing algorithm called Common Multiple Folding (CMF) that can identify periodic signals from an RSS series. CMF has several key advantages including high accuracy and low computation/memory overhead. CMF is based on a technique called *folding* that was first used to search pulsar in the radio noise received by a large radio telescope [81, 85]. We first briefly describe the basic idea of folding and then discuss the details of CMF.

3.2.1 Basic Idea of Folding

Suppose R represents the time series of N RSS samples and $R[i]$ $(i \in [1, N])$ is the RSS magnitude in the ith sampling instance. The objective of folding is to search for a periodic signal with period of P. The series is first divided into smaller sequences with length of P at different starting points (e.g., phases). For each folding operation, the sequences are added together in an element-wise fashion. If the phase of folding happens to align with that of the periodic signal, the magnitude of the sum will be amplified at a period of P while the sum of noise in the series is likely smaller due to their non-periodicity. The sum of folding consists of P elements of R:

$$F_P[i] = \sum_{j=0}^{\lfloor N/P \rfloor - 1} R[i + j \cdot P] \tag{3.2}$$

$F_P[i]$ is referred to as the ith *folding result* and the maximum is referred to as the *folding peak* of period P. It can be seen that the folding operation requires $N - P$ number of additions. When P is unknown, folding can be performed for each possible period and the maximum folding and P can then be found as the period that yields the maximum folding result. In [85], the fast folding algorithm (FFA) was developed to implement the above approach while reducing the redundant additions in the folding of different periods. However, FFA is designed for searching for non-integer periods between P_0 and $P_0 + 1$ and it requires the ratio N/P_0 to be power of 2.

3.2.2 Common Multiple Folding

A key challenge of searching for WiFi beacons from an RSS series is that the period(s) of beacons is not only unknown but also has a wide range. Although the default beacon period setting is 102.4 *ms* on most 802.11 APs, in practical scenarios, the beacon period can range from 60 *us* to 200 *us*, which lends to hundreds of possible beacon settings. As a result, applying folding iteratively for this range incurs high complexity. As discussed in Section 3.1, N could be large due to the high RSS sampling rate required to capture a beacon transmission.

We now present a novel algorithm called *Common Multiple Folding* (CMF) that can *minimize* the total number of additions required to fold on multiple periods. The complexity of CMF is $O(\lg|P| \cdot lcm(P) + N)$ where $lcm(P)$ is the least common multiple (LCM) of the numbers in P. The design of CMF is based on the observation that the folding result of period P can be efficiently computed from that of period Q if Q is an integer multiple of P, *i.e.*, $Q \; mod \; P = 0$. Formally, given folding result F_Q, only $Q - P$ additions are needed to obtain F_P. In comparison, total $N - P$ additions are needed to compute F_P directly from original N RSS samples. For example, Fig. 3.3 illustrates that the folding result of period 6 can be calculated by an additional folding operation on the result of period 12. For instance, the first element in the folding result of period 6 can be computed by a single addition of the the first and seventh elements in the folding result of period

12, *i.e.*, $F_6[1] = F_{12}[1] + F_{12}[7]$. In total, $12 - 6 = 6$ additions are required to fold on period 6 if the folding results of period 12 are already available. In comparison, total $N - 6$ additions would be needed if the folding is directly applied to the original RSS samples.

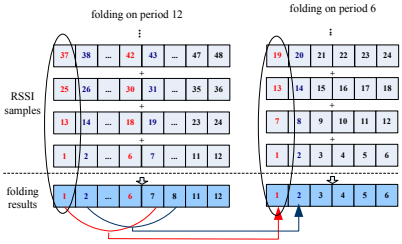


Figure 3.3: The folding operations for period 12 and 6.

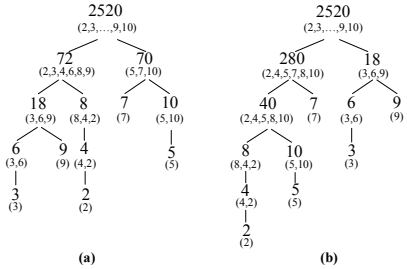


Figure 3.4: Two CMF Trees for folding on periods from 2 to 10.

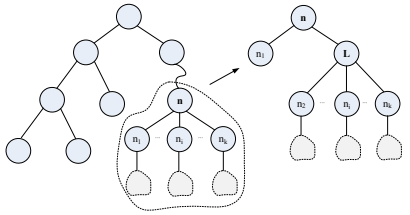


Figure 3.5: Proof of property 1 of optimal CMF tree.

Based on the above example, a promising approach to reducing the computational cost is to first fold on the LCM of all periods in P, and then reuse the results to fold on each of the periods. In order to maximize the utility of intermediate folding results, this idea can be applied *recursively* by partitioning P into subsets and folding on the LCM of all periods in each subset. This process can be naturally encoded by a tree where a node represents a period set and its LCM and all children of the node constitute the partition of the set. We refer to such a tree as *CMF tree*. Fig. 3.4 shows two CMF trees that differ in how to partition the period set at each node.

Once a CMF tree is constructed from a given period set, folding on all periods in the set can be performed by traversing the tree in the breadth-first order and folding on the LCM of each node. For instance, in the left tree in Fig. 3.4, one can first fold on 2520 (by $N - 2520$ additions for total N RSSI samples), and then on 72 (by 2520-72=2448 additions) and 70 (by 2520-70=2450 additions) etc., which results in total $N + 2654$ additions. It can be seen that a similar procedure for the right tree in Fig. 3.4 requires total $N + 2832$ additions. This example shows that the computation cost of a CMF tree depends on how it partitions the period set at each node. An interesting question is how to find the *optimal* CMF tree that yields the least number of additions. We have designed an algorithm that can find the optimal CMF tree [94].

3.3 CFAR Beacon Detector

The output of CMF contains the folding results of all periods in P. The next task of ZiFi is to identify which results correspond to true WiFi beacons. However, this is not trivial due to the following reasons. 1) Non-beacon signals such as WiFi data frames or interference from other RF sources may also exhibit periodicity. The resulted strong folding peaks may be falsely detected as beacons. 2) A beacon frame must compete for the channel with other pending frames and defer its transmissions when the channel is busy. As a result, the transmission times of beacons may become aperiodic leading to detection misses. 3) An RSS sample may participate in the folding of multiple periods. Therefore, the RSS samples of a real beacon signal are essentially noise to the detection of beacons of other periods. We refer to such noise as *cross-period* noise. Fig. 3.6 illustrates cases 1) and 2) in the folding result of a TelosB mote trace. We also used the trace collected by a WiFi sniffer to label each RSS sample of the TelosB mote and identify three types of peaks due to beacons, data frames, and deferred beacons. It can be seen that the deferred beacons often cause a cluster of small folding peaks because of their random backoff delays.

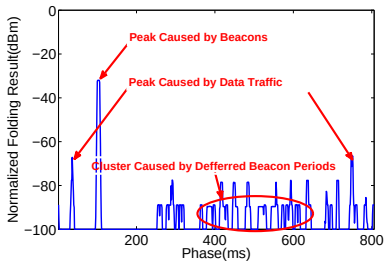


Figure 3.6: The folding peaks of beacons, data frames, and deferred beacons.

We have developed a constant false alarm rate (CFAR) detector [88] to identify WiFi beacons from folding results of CMF. A CFAR detector minimizes the FN rate while satisfying the FP rate upper bound specified by user [88]. There exist fundamental trade-offs between the FN and FP rates of beacon detection. For instance, although using a high threshold to detect folding peaks reduces FPs, it may cause excessive FNs when the folding peaks of real beacons are not strong. CFAR enables a user using ZiFi to specify the FP upper bound based on the mobile devices' energy budget, while allowing ZiFi to automatically minimize the FN rate, i.e., the probability of missing

WiFi connectivity.

A challenge of designing a CFAR detector for our problem is to model the detection FPs and FNs caused by non-beacon signals and 802.11 backoff. Our analysis in [94] showed that the inaccuracy of beacon detection is closely dependent on channel utilization. Intuitively, a busier channel likely reduces the signal to noise ratio in beacon detection due to more interference from periodic noise. At the same time, the likelihood that a beacon frame suffers from backoffs is also higher. Our analytical results in [94] ensure that the beacon detector can use the optimal detection threshold to achieve desired upper bound on FP rates while minimizing the FN rate. We further optimize the beacon detector by adopting a cross-period noise reduction mechanism. Specifically, when a beacon signal is identified, all the RSS values of it are removed before the folding is performed on another period.

The pseudo code of beacon detector is shown in Algorithm 1. It takes as input the RSS samples R, user-specified upper bound on FP rate (denoted by FP), and outputs the periods and beacons that are detected. Initially, the channel utilization U is estimated based on R as follows:

$$U = \frac{|\{R[i] \mid (R[i] \in R) \wedge (R[i] \neq 0)\}|}{|R|} \tag{3.3}$$

In Eq. (3.3), the channel is deemed as busy if the RSS sample has a non-zero magnitude. At step 2, the detector computes the detection threshold α based on U and FP according to our analytical result in [94]. At step 3, the detector runs CMF to perform folding on RSS samples R for all the periods in P. The folding results are stored in $\{F_P\}$ where F_P is the set of folding results of period P. At step 4, the maximum folding result is first normalized by factor P and then compared against the threshold α. We note that the normalization is needed because the number of additions in a folding result is inversely proportional to the period. A folding peak greater than the threshold indicates a successful detection. The period and phase of the maximum folding peak are then used to find the RSS samples of detected beacon. The magnitude of these RSS samples are set to zero at step 9 to reduce the cross-period noise. The above process is then repeated for finding beacons of other periods until the maximum folding peak is smaller than the detection threshold.

As discussed previously, the WiFi interference detection performance is inherently probabilis-

Algorithm 1 Beacon Detector.

Input: R - RSS samples; FP - user-specified FP rate upper bound; P - set of possible beacon periods.
Output: P^* - set of periods of detected beacons, initially empty.

1: Compute channel utilization U using R by Eq. (3.3).
2: Compute threshold α using U and FP by Eq. (3.4).
/*perform folding for all possible periods*/
3: $\{F_P | P \in P\} = Common\_Multi\_Folding(R, P)$.
/*find the max normalized folding result of each period*/
4: $\{F_P^{max} | P \in P; F_P^{max} = P \cdot \max_i F_P[i]\}$.
5: Sort $\{F_P^{max}\}$ in the descending order.
6: **for all** $\{F_P^{max} \mid P \in P\}$ **do**
7: **if** $F_P^{max} \geq \alpha$ **then**
8: $\hat{i} = arg\max_i \cdot F_P[i]$
/*remove RSS values of detected beacons.*/
9: $\forall j \in [0, \lfloor N/P \rfloor - 1],\ R[\hat{i} + j \cdot P] = 0$
10: $P^* = P^* \cup \{P\}$
11: *goto 1*
12: **else**
13: *return*
14: **end if**
15: **end for**

tic due to several error sources: the periodicity of non-beacon WiFi signals, beacon back-off delays, and cross-period noise. We have modeled the impact of the first two factors on the detection performance. We only give the results of the FP model in this dissertation, the detailed analysis of the FP model and the FN model can be found in [94].

The overall FP rate (denoted by FP) is equal to the probability that a FP occurs for any period $P \in \mathscr{P}$ where $\mathscr{P}$ is the period set searched by ZiFi:

$$FP = 1 - \prod_{P \in \mathscr{P}} (1 - f(P, \alpha)) \tag{3.4}$$

$$f(P, \alpha) = 1 - \left(1 - \sum_{k=\lfloor \frac{\alpha}{P} \rfloor}^{N/P} \binom{N/P}{k} U^k (1-U)^{N/P-k} \right)^P \tag{3.5}$$

where N is the number of RSS samples collected, and U is the average channel utilization ratio. We did not model the impact of cross-period noise, as it is effectively mitigated by iteratively removing RSS samples of detected beacons.

For a given FP upper bound, detection threshold α can be easily computed by Eq. (3.4). To reduce computation overhead, we discretize the possible FP range, compute corresponding α values offline, and store them in a table for online lookups. We note that a small FP bound is often desired in order to reduce unnecessary WiFi NIC wake-ups. Therefore, the storage cost of α table is small.

CHAPTER 4

WIRELESS LAN DISCOVERY

4.1 Introduction

In recent years, WiFi networks have enjoyed an unprecedent penetration rate. In particular, they are increasing deployed to provide Internet access in mobile environments. However, due to the limited coverage, existing WiFi infrastructure is only capable of providing intermittent connectivity for the users with high mobility. WiFi-enabled devices (e.g., laptops, PDAs, and smartphones) must actively discover new WiFi access points (APs) once they leave the coverage of current network. However, this approach wastes the precious energy of mobile devices due to excessive listening and scanning operations of WiFi network interface cards (NICs).

In this chapter, we propose a system called *ZiFi* for discovering the availability of WiFi coverage for mobile users. The design of ZiFi is motivated by the fact that low-power radios such as ZigBee and Bluetooth often not only physically collocate with WiFi NICs but also share the same open radio frequency band with them. Leveraging the inter-platform interference caused by such coexistence, ZiFi enables ZigBee radios to identify the unique interference signatures generated by WiFi signals. As a result, a mobile device can use a ZigBee radio to detect the existence of WiFi APs in a purely passive manner, and only wakes up the WiFi NIC when WiFi connectivity is available. To capture WiFi interference signatures, ZiFi utilizes the received signal strength (RSS) indicator available on ZigBee-compliant radios. However, we observed that the statistics of WiFi RSS samples, such as power magnitude, time duration, and inter-arrival gap, exhibit surprising resemblance with those of other RF sources, and hence provide little hint about the existence of WiFi. Motivated by this observation, ZiFi is designed to search for 802.11 *beacon frames* in RSS samples. Periodic beacon broadcasting is mandatory in WiFi infrastructure networks and hence provides a reliable means to indicate WiFi coverage. However, beacons are extremely scarce in normal WiFi

traffic as hundreds of data frames are likely transmitted between two beacon instances. Without being able to decode incoming signals, finding beacon frames in RSS samples is like finding a needle in a haystack. To address this challenge, ZiFi adopts the digital signal processing (DSP) and stochastic signal detection techniques described in Chapter 2 to reliably identify the periodic interference patterns caused by WiFi beacon frames.

We envision the approach of ZiFi to be increasingly feasible as more mobile devices are equipped with both low-power and high-power NICs that work in the same open radio spectrum. For instance, numerous ZigBee modules [32] have USB interface and hence can be easily connected to WiFi-enabled laptops. Several cell phone vendors (e.g., Nokia and Pantech & Curitel) also provide smartphones [21] with built-in ZigBee interface or ZigBee modules [96] that can be connected to smartphones (e.g., through miniSD interface). ZiFi can also be easily implemented on other platforms (e.g., some Bluetooth radios [1][23]) that offer the RSS sampling interface.

4.2 Design of ZiFi

The design objectives of ZiFi include the following: 1) *High accuracy.* We characterize the accuracy of AP detection using *false positive (FP)* and *false negative (FN)* rates. In particular, FPs falsely trigger the wake-up of NICs leading to energy waste while FNs mean the misses of opportunities of WiFi connectivity. 2) *Low delay.* This is of particular importance for mobile environments. For instance, recent war driving experiments in Boston metropolitan area [37] showed that the median duration of WiFi connectivity at vehicular speeds is only 13 seconds. Fast WiFi discovery is thus required to utilize such short connectivity windows. 3) *Low overhead.* Due to the resource constraints of mobile devices, the computation and memory overhead in WiFi discovery must be small.

ZiFi is designed for two different types of platforms to discover WiFi APs: the platforms (e.g., smartphones) that have both built-in ZigBee and WiFi interfaces, and the platforms that can connect a WiFi node with an external ZigBee node. Fig. 4.1 shows two platforms of the second type on which ZiFi has been implemented. Fig. 4.1 (a) is a Nokia N73 mobile phone that

integrates a ZigBee module through the miniSD interface. Fig. 4.1 (b) is an ASUS Linux netbook that connects a TelosB mote (equipped with a ZigBee-compliant CC2420 radio [74]) through the USB interface.

(a) (b)

Figure 4.1: Two different platforms on which ZiFi has been implemented. (a) is a Nokia N73 mobile phone and a ZigBee card integrated via the miniSD interface. The top of figure shows front and back views of the ZigBee module and its main components. (b) is an ASUS netbook and TelosB mote integrated via the USB interface.

Fig. 4.2 shows the system architecture of ZiFi. The *RSS sampler* reads the built-in received signal strength indicator (RSSI) register of ZigBee radio at a designated frequency. The RSS samples are then processed by a *RSS shaper* that adjusts the RSS values to mitigate noise (e.g., the data frames) in the beacon detection. The shaped RSS samples are then processed by the Common Multiple Folding (CMF) algorithm. CMF is a digital signal processing algorithm that amplifies the periodic signals in RSS samples. A key advantage of CMF is that it can minimize the cost of amplifying unknown signals whose possible periods lie in a wide range. The amplified RSS samples are fed into a *constant false alarm rate (CFAR) [88] beacon detector* that classifies a periodic signal as genuine WiFi beacons if its amplitude exceeds a threshold. By adopting a theoretically derived threshold, the beacon detector can minimize the false negative (FN) rate while satisfying the user-specified upper bound on false positive (FP) rate. Finally, if WiFi beacons are detected, the *radio controller* turns on the WiFi NIC. In this chapter, we also present an analytical framework that models the FN and FP rates of beacon detection based on the utilization ratio of wireless channel. The utilization ratio is measured from RSS samples by the *channel profiler*. The

analytical FN and FP models guide the selection of optimal detection thresholds for ZiFi's beacon detector.

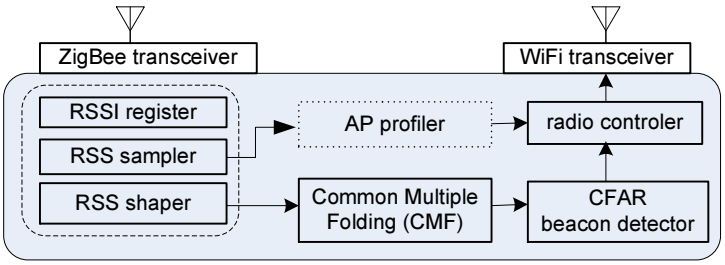


Figure 4.2: System architecture of ZiFi.

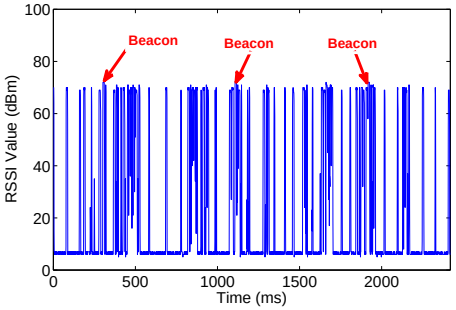


Figure 4.3: The RSS samples collected by a CC2420 radio when a nearby 802.11 AP transmits.

As discussed above, ZiFi utilizes energy sensing through the RSSI of ZigBee radio to detect the existence of WiFi APs. ZiFi can be easily implemented on other radio platforms that provide the RSSI interface. For instance, a few existing Bluetooth radios [1][23] provide RSSI although it is not a mandatory feature in Bluetooth standard.

A challenge in the design of ZiFi is that WiFi APs operate on unknown channels. Running ZiFi on each of the 11 802.11 channels in 2.4 GHz band would lead to significant detection delay. Due to the overlap between 802.11 and 802.15.4 channels [93], we found that running ZiFi on four 802.15.4 channels (1,5,8,11) can reliably detect the APs running on all 11 802.11 channels.

ZiFi is able to find multiple beacons from different APs. Thus an interesting issue is how to aid WiFi NIC to choose the best AP to associate with. The quality of an AP depends on several factors such as channel utilization and SNR. ZiFi is able to obtain such information during the detection processes to assess the quality of each AP, through a module called *AP profiler*.

AP profiler assess two quality metrics of AP solely from Zigbee RSS series: channel utilization rates, and SNR of APs. These two metrics largely determine the achievable throughput between the client and the AP. When the application can tolerate certain communication delay, user turns on WiFi NIC only if the quality of detected APs is beyond a certain threshold. This scheme reduces the energy consumption since the resulted high throughput allows data to be transmitted in shorter

time. Therefore, *AP profiler* allows user to trade off between communication delay and energy consumption.

Channel utilization. Channel utilization indicates the percentage of occupied time slots on an 802.11 channel. Due to the sharing nature of wireless channels, only non-occupied time slots on a channel can be utilized by clients. As discussed in Section 3.3, ZiFi computes the channel utilization as the ratio of the number of RSS samples that have signal strength above the noise threshold, to the total number of RSS samples. The channel utilization is computed by *AP profiler* for each channel scanning.

Beacon SNR. SNR indicates the quality of the signals received by the client. It largely determines the maximum throughput between client and AP when the channel is vacant. ZiFi periodically assesses the noise level by averaging the RSS values in the RSS series when there is no activity on the channel. The signal strength of beacons is obtained during the beacon removal process after an AP is detected. Since the RSS sampling usually spans over multiple beacons periods, one RSS series always contains multiple beacon signals of the same AP. The strengths of these signals are averaged to mitigate signal fluctuations caused by wireless channel dynamics, such as multipath fading.

4.3 Experimentation

4.3.1 Experimental Setup and Methodology

We implemented ZiFi on two platforms: ASUS Linux netbook integrating a TelosB mote through the USB interface, and Nokia N73 smartphone integrating a ZigBee card through the miniSD interface. The CMF algorithm is implemented in Mablab on netbook and in C++ on Nokia N73. On both platforms, the RSS sampler of ZiFi is implemented in ZigBee module and all other components run on netbook or Nokia N73.

Our experimental testbed consists of four 802.11g APs, four Linux-based 802.11 netbooks, two TelosB motes equipped with CC2420 radios, and a Nokia N73 smartphone. The performance of

ZiFi depends on both the characteristics of WiFi APs (e.g., data rate and transmit power) and user traffic (e.g., workload). In our experiments, the user traffic is generated from a high-fidelity Internet traffic generator called D-ITG [7] that runs on our APs. D-ITG has several advantages over the existing traffic generators such as the capability of generating multiple simultaneous flows from different protocols. Empirical results showed that D-ITG can reproduce realistic traffic patterns under a wide range of network settings [7]. The use of D-ITG allows us to evaluate ZiFi in comprehensive WiFi and traffic settings, which would be impossible for using particular operational WiFi deployment. We note that several large-scale WiFi traces (e.g., the SIGCOMM [25] and OSDI [6] traces) are publicly available. However, they are collected under particular network settings. Moreover, it is extremely difficult to replay the collected WiFi traces with high fidelity.

4.3.2 Detection Accuracy

We evaluate the detection accuracy of ZiFi using a Linksys WRT54G2 router as AP, and two ASUS Linux netbooks as clients. ZiFi is executed on one netbook which connects with a TelosB mote via USB. The traffic generated by another netbook contains one TCP stream and one UDP stream. The length of frames is uniformly distributed, from 5 to 1400 bytes for TCP and from 50 bytes to 1400 bytes for UDP. We vary the average frame transmission rate to control the channel utilization. The distance between AP and ZiFi node is 3 meters. The network traffic is logged as ground truth for micro-scale analysis of ZiFi performance. The experiments were conducted in a residential environment. The AP uses channel 1 and both AP and client transmit at the default power level. The length of AP beacon period is configurable at a step of 1.024 *ms*. We varied the period length and observed no obvious performance variation of ZiFi. We used a fixed period of $96 \times 1.024 = 98.304$ *ms* throughout all experiments. However, as this setting is unknown to ZiFi, the CMF algorithm of ZiFi searches for the beacon period within the range of $(60 \sim 120) \times 1.024$ *ms*.

Comparison with other signal processing approaches. Autocorrelation and FFT are two signal processing algorithms widely used to detect periodic signals. We now compare the performance of

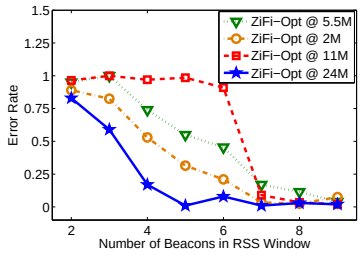


Figure 4.4: Detection error rate vs. number of beacon periods.

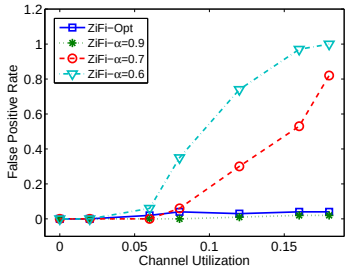


Figure 4.5: Detection error rate vs. ZigBee channel utilization.

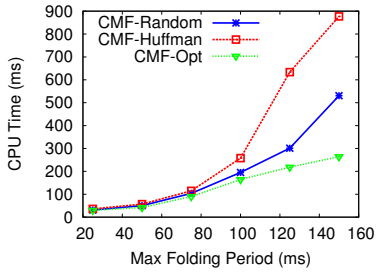
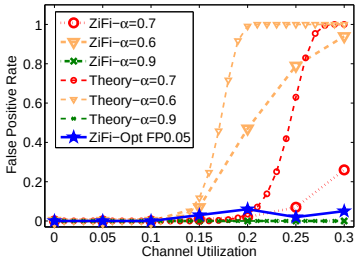
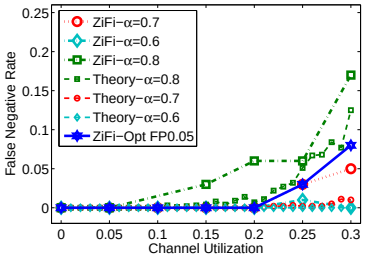


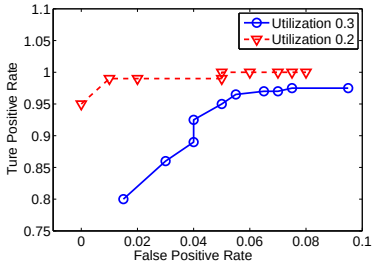
Figure 4.6: CPU overhead of CMF on N73 smpartphone.



(a) FP rate vs. channel utilization ratio.



(b) FN rate vs. channel utilization ratio.



(c) True positive rate vs. FP rate.

Figure 4.7: Detection performance of ZiFi.

them against that of ZiFi. Fig. 3.2 shows the results when they are applied to 10000 RSS samples (i.e., 1.22 second) of the traffic. The AP modulation rate is set to 2 Mbps. As can be seen from Fig. 3.2 (a) and (b), the results of Autocorrelation and FFT can clearly reveal the beacons when the data rates are 30 Kbps and 100 Kbps, respectively. However when the data rates increase to 180 Kbps and 260 Kbps, respectively, both of them fail to detect beacons. These results show that Autocorrelation and FFT cannot reliably identify the existence of WiFi networks. In contrast, ZiFi successfully detects beacons under all settings. Fig. 3.2 (c) shows the folding results of ZiFi for 400 Kbps where the beacon peaks can be clearly identified.

Impact of RSS window size. The size of RSS window used by ZiFi is a critical design parameter as it directly determines the detection delay. Fig. 4.4 shows the detection error rates of ZiFi with different RSS window sizes. The AP cycles through four modulation rates during transmission while the channel utilization ratio is always tuned to 30%. Each data point is the average of 5 runs. For each run, ZiFi carries out the detection for 40 times and the error rate is computed as the probability of failing to detect a beacon or falsely detecting a non-beacon signal, i.e., the sum

of FN and FP rates. Since the in-air time of data frames transmitted at 11 Mbps is very close to that of beacon frames, only a few RSS samples of data frames are removed by the RSS shaper, causing significant noise in the folding results. However, even in this worst case, the error rate of ZiFi quickly decreases to near-zero when the RSS window contains more than only 7 beacons. For instance, when 8 beacon periods of RSS samples are used (which corresponds to a total delay of $8 \times 96 \times 1.024 = 786.4$ *ms*), ZiFi's average error rate under four channel rates is only 4.8%.

Impact of ZigBee interference. The RSS samples gathered by ZiFi may contain the transmissions of other devices operating in the open radio spectrum. These transmissions create noise in WiFi beacon detection. In particular, the RSS samples can be easily contaminated by the traffic of peer ZigBee nodes due to the similar signal in-air time. Since decoding is disabled during RSS sampling to decrease the possibility of RSS sampling interrupting, such noise cannot be eliminated. We now evaluate the impact of such interference from peer ZigBee nodes. In the experiment, a pair of TelosB motes transmit on an overlapping channel. The frame sizes are uniformly distributed between 14 and 74 bytes. The in-air time of these frames has a significant overlap with that of WiFi traffic. The data rate of transmission is varied from 1.35 to 27.1 Kbps to obtain different channel utilization ratios. Three ZiFi variants are used as baselines for comparison. 'ZiFi-α=x' refers to the implementation of ZiFi where the detection threshold is manually set to be $\alpha \cdot N/P$ where $\alpha \in (0,1]$, N and P are the total number of RSS samples and the real beacon period, respectively. 'ZiFi-opt' refers to the default implementation of ZiFi that computes the threshold based on an 0.05 FP upper bound. Fig. 4.5 shows that ZiFi-opt yields near-zero false positives. In contrast, two ZiFi variants falsely classify more ZigBee signals as WiFi beacons when the channel workload is higher. The main reason is that ZigBee traffic contains more periodic signals under heavier traffic load, which results in many folding peaks. ZiFi-opt automatically chooses high detection thresholds to avoid such false positives and yield similarly performance as 'ZiFi-α=0.9' which has a manually set high threshold.

Impact of traffic workload. Our objective of evaluation in this experiment is three-fold. First, we test the FP and FN rates of ZiFi under various settings of channel utilization ratio and bit

rate. Second, we compare the experimental results with analytical result presented in [94]. Third, we plot the receiver operating characteristic (ROC) [88] curve of ZiFi. ROC characterizes how true positive (TP) rate varies with FP rate and is widely adopted for evaluating the capability of detection systems.

The maximum channel utilization ratio in our evaluation is set to 30%. We note that real-world WiFi deployments usually have low channel utilization ratio. Our analysis of over 10^4 seconds of traces collected at OSDI 2006 [6] and SIGCOMM 2008 [25] show that the channel utilization ratios (computed per second) have a mean of 7.58% and 0.81%, and median 7.2% and 0.3%, respectively. This result is also consistent with the recent finding [49] that significant white space exists in real WiFi traffic. Fig. 4.7(a) and (b) show the FP and FN rates under different channel utilization ratios. It can be seen that ZiFi variants with fixed detection thresholds yield poor FP or FN rate. For instance, although a low threshold (e.g., 0.6) has a near-zero FN rate, it leads to extremely high FP rates when channel is heavily loaded. This is because the folding peaks of noise (data frames) become higher and many of them exceed the low threshold. However, when the threshold is set to 0.9, although the FP rate is low, many beacons are missed. In contrast, ZiFi-opt can achieve both satisfactory FP and FN rates by automatically adjusting the threshold based on channel workload. Under all settings, ZiFi-opt has a FP rate lower than the preset upper bound 0.05 while achieving low FN rates. It can be seen from Fig. 4.7(b) that the theoretical prediction matches the experimental FN under all settings. However, Fig. 4.7(a) shows a considerable gap between theoretical and experimental FP rates when the channel utilization is high ($>$). This is due to two reasons. First, ZiFi implements an RSS shaper that can remove some noise, e.g., the RSS samples that are likely data traffic (see Section 3.1). However, the impact of RSS shaper is not modeled in our FP analysis. Second, our theoretical FP analysis is based on a uniform channel utilization model where the probability at which any slot is busy is constant. However, the data traffic under this model yields better periodicity than reality because the burstiness [49] of real WiFi traffic is not considered. As a result, the theoretical FP rate is a pessimistic estimation of real FP rate. Improving the accuracy of our theoretical FP model by accounting for the above factors is

left for future work.

Fig. 4.7(c) plots the ROC curves of ZiFi. We vary the user-specified FP upper bound from 0.01 to 0.46 at a step of 0.05, and calculate the true positive (TP) and FP rates for each setting. It can be seen that ZiFi achieves a good TP rate if the allowable FP rate is above 4%. We can also see that ZiFi has a desirable configurability and allows a user to achieve trade-offs between FP rate and TP rate. For instance, a mobile device may set a higher FP bound to maximize the opportunity of finding WiFi networks while setting a lower FP bound to reduce the number of NIC wake-ups for energy conservation.

4.3.3 Computational Overhead

We measure the CPU overhead of ZiFi on Nokia N73 smartphone in this experiment. 80K RSS samples are used for finding beacons whose periods lie between 25 *ms* and 150 *ms*. We note that this range can be properly scaled to obtain other period ranges. We use two CMF variants for performance comparisons. They differ from CMF-opt in how to create a new node. CMF-Random randomly choose children from a given node set while CMF-Huffman chooses the two nodes with the minimum values. Fig. 4.6(b) shows the CPU overhead of folding on the trees found. We can see that the CPU usage for both CMF variants increases sharply when the period range exceeds 100 *ms*. In contrast, CMF-opt has significantly lower overhead. The CPU time of running CMF-opt under all settings is up to 200 *ms*.

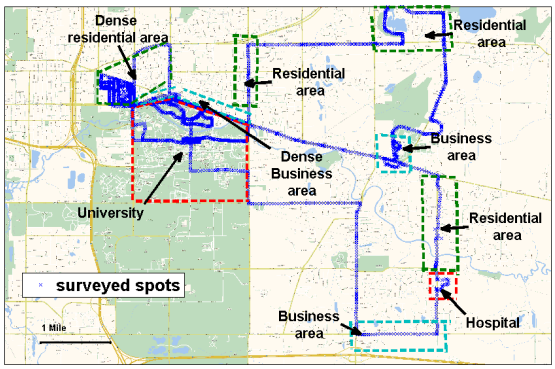


Figure 4.8: The site survey covers multiple regions of East Lansing, Michigan.

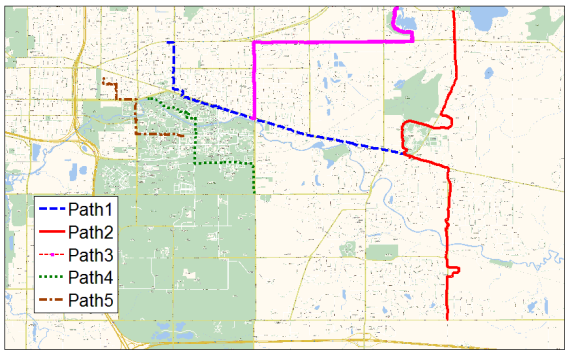


Figure 4.9: Five paths travel through different regions of the city.

		Channel utilization			
		0.1	0.2	0.3	0.4
SNR	50dB	17.3Mbps	15.3Mbps	13.6Mbps	11.1Mbps
	20dB	16.0Mbps	14.9Mbps	12.4Mbps	10.8Mbps
	10dB	12.1Mbps	10.8Mbps	9.2Mbps	8.4Mbps
	5dB	5.1Mbps	4.3Mbps	3.8Mbps	3.1Mbps
	2dB	N/A	N/A	N/A	N/A

Table 4.1: Resulted throughput under different SNR and channel utilization combinations.

4.3.4 Effectiveness of AP Profiling

We evaluate the effectiveness of the AP profiling module in this section. A testbed consisting of one AP and two laptops is adopted to test the throughput under different combinations of SNR and channel utilization measured by ZiFi. The two laptops, which we refer as laptop A and laptop B, are associated with the AP during experiments. A traffic generator running on laptop A creates a desired utilization rate on the channel, emulating the traffic generated by already associated users. Laptop B, which is the client that attempts to connect to the network, collects RSS samples using a TelosB on the same channel and computes the SNR of the AP and the channel utilization rate using the methods described in Section 4.2. We manually tune the parameters of the traffic generator on laptop A as well as adjusting the distance between laptop B and the AP to create a certain channel utilization and SNR combination. We then measure the throughput between the AP and laptop B using iPerf[13], which is a widely used traffic generator and bandwidth measurement tool. Tab. 4.3.4 shows the results of our experiments.

As expected, the result shows that the measured throughput monotonically decreases with SNR and channel utilization rate. Note that the throughput cannot be measured for combinations with SNR less than 2 dB, since the laptop failed to reliably associate with the AP under such low SNR in our test. We observed a almost 6X difference in throughput (17.3Mbps vs 3.1Mbps). This significant throughput variance implies that user can save more energy by only initiating association when the achievable throughput is high enough. This is because that higher (lower) throughput means transmitting/receiving the same amount of data takes less (more) time, which reduces (in-

creases) energy consumption of the WiFi interface. Users can exploit this result by adopting a rule that filters out the APs with low throughput during ZiFi detection process. Even if the user prefers to associate to APs whenever is possible (e.g., to avoid delay), the AP profiling module can still help filter out those APs with extremely low SNR that would otherwise fail association and waste the energy.

4.3.5 Performance in Mobile Scenarios

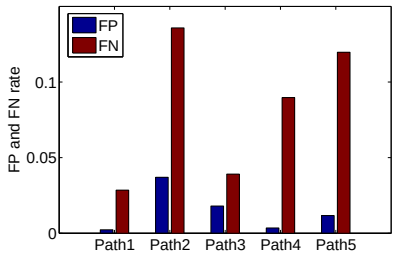


Figure 4.10: Accuracy of ZiFi detection in mobile scenario.

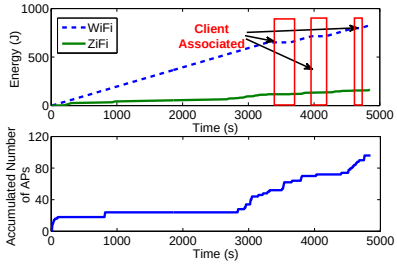


Figure 4.11: Energy consumption comparison over time on path 4.

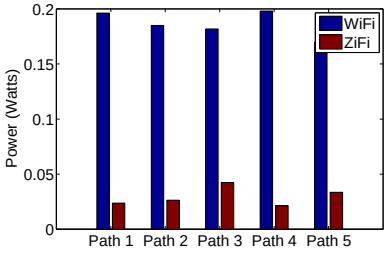


Figure 4.12: Power consumption comparison on all paths.

In this section, we use trace-driven simulations to study the detection accuracy and the power consumption of ZiFi in mobile scenarios. We collected the data trace in a large scale site survey around the City of East Lansing, Michigan. The footprint of the surveyed area is approximately 25 square miles, covering residential areas, business areas, Michigan State University campus, and etc. During the site survey, a laptop recorded all the overheard WiFi traffic on channel 1, 6 and 11 every 5 seconds, spending 1 second on each of the three channels. A GPS attached to the laptop provides geographical coordinates. We conducted the site survey on a motor vehicle for a period of 6 hours over a distance of 51 miles. The resulted data trace contains 1402 APs, with signal strength varying between -30 dBm and -95 dBm. For possible privacy concerns, we anonymized all the IP addresses and MAC addresses. The actual coverage of our site survey is shown in Fig. 4.8.

We study the mobile performance of ZiFi with simulations driven by the collected data traces. We first create paths emulating the routes of users using a Matlab graphical mapping tool. We

Path	Length	regions passed	APs
1	11,201 m	dense business	132
2	7,075 m	residential, business	221
3	6,840 m	residential	154
4	4,839 m	university	95
5	3,597 m	university, dense residential	176

Table 4.2: Statistics of the five paths.

create five non-overlapping paths traveling through different sections of the city with different WiFi coverage conditions, as shown in Fig. 4.9. The statistics of the five paths are given in Tab. 4.3.5. We then generate WiFi coverage maps of the five paths based on our survey data. For the WiFi detection algorithm, a location is WiFi-covered if the maximum signal strength of the received beacons is above -85 dBm[1]. As no Zigbee data trace is available, we use a method called RSS resampling to generate the Zigbee RSS series according to the collected WiFi data traces. This method essentially simulates the sampling operation of ZigBee radios. The RSS magnitude is set to either a fixed value when the sampling time falls within a beacon duration or zero otherwise.

Using the coverage map generated by WiFi detection algorithm as ground-truth, we calculate the accuracy of ZiFi detection algorithm. Fig. 4.10 shows the resulted FP and FN rates. As can be seen, the FP and FN rates of all paths fall below 0.04 and 0.14, respectively, while the average FP and FN rates are 0.014 and 0.08, respectively. Interestingly, we observe that the moving speed of the client is strongly correlated with the FN rate. Path 2 and 1 achieve the highest and lowest FN rates among five paths, respectively. It is worth noting that, the moving speeds of the vehicle during the site surveying along the two paths are also the highest and lowest. Path 2 which travels through a large residential area, had little road traffic, resulting in the highest possible moving speed during our site survey. Path 1 which is across the downtown of East Lansing, had heavy road traffic. This leads to a very slow driving speed. This observation likely indicates that the moving speed of the client has significant impact on the resulted FN rate. The client traveling at high speed covers a

[1]Our experiments indicate that a reliable association usually occurs when the signal strength of the beacon is above -85dBm.

long distance during one ZiFi scanning window, in which some beacons were not reliably captured due to signal fading. This lowers the folding peak, causing the miss-detection of the beacons. We also observe the correlation between FP rate and AP density in the area. The three paths (2, 3, 5) traveling across residential areas have the highest AP density, and they achieve higher FPs rate than the other two paths. This is possibly because that high density of APs usually leads to heavy WiFi traffic, which increases FP rate(as shown in the results in Section 4.3.2).

We then estimate the energy consumption of ZiFi and the default 802.11 WiFi detection algorithms along the five paths. We measure the parameters (current consumption and scanning duration) for computing the energy consumption offline, using a TelsoB mote and an EeePC equipping with an Atheros AR9270 WiFi NIC. According to our measurements, we set the active current of WiFi and ZiFi to be 100 mA and 20 mA, respectively, and the scanning duration of WiFi and ZiFi are set to 4s and 2s, respectively. Both of the two detection algorithms are executed periodically every 10s. In ZiFi, a WiFi scanning only occurs when a positive scanning result from ZigBee radio is returned. We refer to the WiFi scanning triggered by a positive ZigBee scanning result as a *follow-up WiFi scanning*. In both algorithms, when WiFi driver detects a nearby AP with signal strength higher than -85 dBm, the client automatically associates to this AP until the signal strength of the AP drops below -85 dBm. When associated, we assume that the AP scanning performed by WiFi does not incur additional energy cost. We also assume that the client can roam from one AP to another if the two APs have overlapping coverage.

Fig. 4.11 shows the detection energy consumption over time on path 4. The upper sub-figure shows the energy consumed by the two detection algorithms, while the lower sub-figure shows the accumulated number of APs detected over time. We also marked the time period when the client associated with APs. As can be seen, the default WiFi detection algorithm consumes roughly five times of the energy of the ZiFi detection algorithm. We can also clearly observe that the energy consumption of WiFi detection algorithm increases linearly when the client is not associated with an AP. This is because the WiFi detection algorithm is performed periodically and every detection process consumes a constant amount of energy. The energy consumption of ZiFi largely depends

on the AP density that can be computed as the slope of the curve on the lower sub-figure. With higher AP density, more positive results are returned by ZiFi detection algorithm, which triggers more follow-up WiFi scans.

Fig. 4.12 shows the average power consumption of clients on all paths. We can see that, ZiFi detection algorithm consumes approximately one-fifth the power of WiFi detection algorithm in all cases. We notice that clients have different power consumptions on the five paths. Due to higher AP density, on path 2, 3 and 5, the clients spend more time in associated state, leading to lower power consumptions on AP detection. However, since ZiFi detection algorithm returns more positive results on these three paths than the other two paths due to denser AP deployments, more follow-up WiFi scans are triggered. This leads to higher detection energy consumption on the client managed by ZiFi.

CHAPTER 5

WIFI NETWORK PERFORMANCE MONITORING

5.1 Introduction

In the last decade, WiFi networks have enjoyed a phenomenal penetration rate, making them an important communication infrastructure for pervasive computing applications. However, compared with wired LANs, WiFi suffer significantly higher level of *spatial* and *temporal* performance variability. Due to the broadcast nature of wireless channel, signal propagation are susceptible to environmental conditions. As a result, end-users often experience highly variable signal quality. To diagnose such transient service degradation and plan for future network upgrades, it is essential to closely monitor the spatial and temporal performance of a WiFi network as well as to collect the statistics of its users.

The existing WiFi network performance monitoring solutions [39][40][43] rely on 802.11-based listening devices. However, due to the high power consumption of 802.11 radios, the monitoring nodes must be connected to external power supplies (e.g., wall power or desktop computers). This constraint leads to high installation costs and poor spatiotemporal monitoring granularity. In this chapter, we propose a new approach for WiFi monitoring by leveraging distributed cheap off-the-shelf wireless sensors. Our approach is motivated by the fact that an increasing number of low-power wireless technologies such as ZigBee and Bluetooth co-exist with WiFi in the unlicensed radio spectrum[55]. For instance, many low-power wireless sensor platforms adopt ZigBee-compliant radios that operate in the open 2.4 GHz band. These radios are capable of simple spectrum sensing, e.g., sampling the Received Signal Strength (RSS) indicator. When the RSS is measured in a frequency range overlapping with 802.11 channels, it indicates partial power of 802.11 signals and hence provides important hints of WiFi coverage. The power consumption of a ZigBee radio is typically an order of magnitude lower than that of WiFi radio. Fig. 5.1 shows the

measurement of the power consumption of a TelosB mote and a USB WiFi NIC when they work in scanning and sleep modes. Based on these measurements, the expected lifetimes of ZigBee and WiFi nodes are 3 months and 3.3 days, respectively, if they are powered by 2 AA batteries and adopt a 10% duty cycle (i.e., active for 10% of the time). Due to their low power consumption, ZigBee nodes can be deployed in places like corridors and stairs in a large building where no power outlets are available. The capability of sustaining long lifetimes on small batteries makes low-power ZigBee networks an inexpensive solution for monitoring WiFi performance at large spatial and temporal scales.

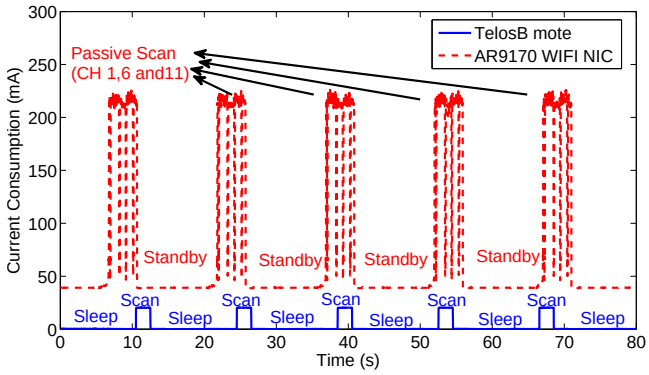


Figure 5.1: Current consumptions of a ZigBee mote and a USB WiFi NIC during scanning and sleeping.

This chapter describes the design and implementation of WizNet – a WiFi performance monitoring system built on 2.4 GHz off-the-shelf ZigBee sensors. By adopting digital signal processing techniques, WizNet automatically identifies 802.11 signals from ZigBee RSS measurements and associates them with wireless access points. To ensure the monitoring fidelity, WizNet accounts for the significant bandwidth difference between ZigBee and WiFi radios. Moreover, the impact of multipath and frequency-selective fading is mitigated by exploiting the wireless spatial diversity through multi-sensor fusion. WizNet adopts a simple yet accurate linear estimator derived from a signal propagation model to infer the access points' signal to noise ratio (SNR). WizNet also measures the channel utilization rate from RSS series, which faithfully indicates the congestion level of wireless channels. The measured SNR and channel utilization rate can be used to predict the WiFi network throughput. Moreover, WizNet detects unauthorized APs (rogue APs) by analyzing

the temporal signatures of 802.11 beacons. Lastly, WizNet is able to identify 802.11 AP scans and classify device models based on their RSS signatures, hence can be deployed in the areas with little or no WiFi coverage to collect statistics of potential users. We have implemented WizNet in TinyOS 2.x and extensively evaluated its performance on a wireless testbed. Our results over a period of 140 hours show that WizNet can accurately capture the spatial and temporal performance variability of a large-scale production WiFi network.

5.2 Background and System Overview

5.2.1 802.11/802.15.4 Spectrum Sensing

Both 802.11 and 802.15.4 technologies work in the unlicensed radio spectrum. In particular, 802.11 standards define 11 channels from 2.412 to 2.462 GHz and 802.15.4 defines 16 channels from 2.410 to 2.480 GHz, which results in a large overlap between the channels of 802.11 and 802.15.4. When operating in overlapping channels, 802.11 and 802.15.4 radios can interfere with each other [55]. Using a built-in register called the Received Signal Strength Indicator (RSSI), ZigBee radios can sense the power of signals emitted by nearby WiFi devices although they cannot demodulate WiFi signals. However, there exists a significant gap between the bandwidths of ZigBee and WiFi radios, which are 3 MHz and 22 MHz, respectively. As a result, the RSSI of ZigBee can only sense the signal power distributed in a fraction of WiFi bandwidth.

5.2.2 Design Objectives

Our goal is to use ZigBee radios as sensors to measure the SNR of 802.11 transmissions, estimate the channel utilization rate, collect client statistics at a set of designated locations, and detect rogue APs. SNR indicates the quality of wireless coverage at a location and has been widely adopted as a metric to characterize the spatial performance of WiFi deployments [77][59]. Channel utilization rate describes how busy the wireless channel is. SNR and channel utilization can be used to infer other important WiFi network performance metrics like throughput. WizNet can also discover

rogue APs that are deployed without the authorization of network administrators. Rogue APs may lead to security breach since they can be exploited by third parties to access the secured networks. WiFi user statistics such as the number of potential users and 802.11 device models provide important information for future network upgrades.

Powered by small batteries, WizNet sensors are inexpensive and easy to install. These features make WizNet ideal for monitoring WiFi performance at large spatiotemporal scales. However, as a signal-level monitoring tool, WizNet is not designed to diagnose packet-level performance and security issues between AP and WiFi clients. Therefore, WizNet is mainly targeted to complement, instead of competing with, existing performance assessment tools based on 802.11 radios. In particular, WizNet sensors can be deployed densely in an ad hoc manner to assist network operators in rapidly locating performance issues of large-scale enterprise WiFi networks, and then integrating with 802.11-based network analysis tools for further packet-level diagnosis.

5.2.3 System Architecture

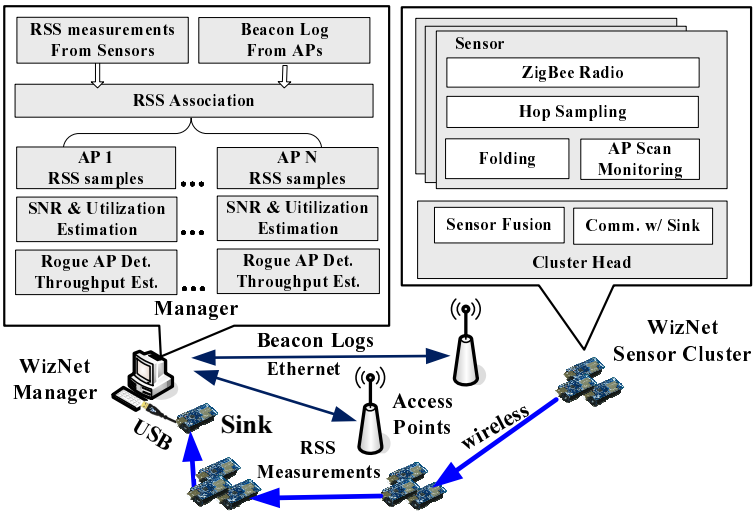


Figure 5.2: System Architecture of WizNet.

Fig. 5.2 shows the architecture of WizNet. WizNet consists of a *manager* computer and a number of ZigBee node clusters, referred to as sensor clusters, which are scattered around the WiFi deployment region. Each cluster is composed of 2 or more closely placed sensor nodes. Wiz-

Net sensors form a possibly multi-hop wireless network whose sink is connected to the manager computer through USB. A set of locations are preselected as *monitoring spots* at which the performance of WiFi networks and user statistics are monitored. The network operators may choose monitoring spots based on user traffic and building floor plans.

WizNet sensors periodically sample the RSS from their radios. To cope with the bandwidth difference between WiFi signal and ZigBee receivers, instead of sampling at a fixed frequency, WizNet introduces a novel technique called *hop sampling*, which samples the signal at different frequency bands and combines the results. Since the resulted signal strengths are derived from a wide bandwidth, the effect of frequency-selective fading is greatly alleviated. Then the RSS measurements are processed through a digital signal processing (DSP) algorithm called *folding* which we describe in Chapter 2. Folding identifies the periodic 802.11 beacon frames from RSS measurements, which are then transmitted to sensor cluster head through wireless links.

The cluster head jointly processes sensor readings through the *sensor fusion* module. By fusing the RSS measurements, WizNet exploits the spatial diversity of different sensors and reduces the impact of multipath fading. Hop sampling, folding, and sensor fusion together enable WizNet to reconstruct the WiFi signal energy distribution based on ZigBee RSS measurements. WizNet also monitors the number of active AP scans from 802.11 client devices and classifies the device models in the areas without WiFi coverage, based on the unique signatures of AP scans of different 802.11 clients.

The WizNet manager implements *RSS and AP association*, *SNR and channel utilization estimation*, and *performance estimation*. First, the manager collects a small amount of information about beacon frames logged by WiFi APs, and then jointly processes them with the beacon RSS measured by sensors through a *cross-correlation* algorithm. The algorithm associates each AP with the RSS measurements of its beacons, without incurring the high overhead of precise time synchronization between APs and WizNet sensors. After associating RSS samples with APs, WizNet manager estimates the SNR of each AP and the channel utilization rates at monitoring spots. WizNet employs a simple yet accurate linear estimator derived from a signal propagation and re-

ception model to estimate the WiFi SNR. The manager computes channel utilization rate according to the logged channel activities in the RSS series. Finally, the manager estimates the throughput between local WiFi clients and the monitored APs, and detects rogue APs.

5.3 Design of WizNet Sensor

5.3.1 RSS Hop Sampling

To achieve high monitoring fidelity, WizNet should ensure that ZigBee RSS measurements accurately reflect the signal quality of 802.11 transmissions. However, RSS only measures the power of the portion of signal that lies in the receiving bandwidth. Due to the narrow bandwidth, ZigBee RSS measurements are usually highly susceptible to prevalent frequency-selective fading caused by the heavy indoor multipath effect.

To address this issue, we employ a novel technique called hop sampling. Hop sampling periodically changes the center frequency of ZigBee receivers when measuring RSS, which enables ZigBee receiver to sample RSS from a much wider bandwidth. WiFi beacon signal is always modulated at the lowest bit rate by the DSSS scheme which spreads the baseband signals to a 22 MHz bandwidth. WizNet divides the 22 MHz 802.11 channel into 7 adjacent non-overlapping 3 MHz sub-channels. During hop sampling, sensors sweep through these sub-channels in order and stay at each sub-channel for one 802.11 beacon period (typically 102.4 *ms*). During the dwell time on each sub-channel, the sensor samples its RSSI register at 8.192 kHz which is sufficient for capturing WiFi beacon frames[36][94]. This process takes less than one second during which the wireless channel is usually stable. Since these sub-channels are non-overlapping and adjacent, the RSS of the WiFi signal can be calculated by summing up the group of RSS values measured from these sub-channels.

5.3.2 RSS Folding

The RSS samples may contain signals of other 2.4 GHz wireless devices such as ZigBee, Bluetooth, or cordless phones. WizNet needs to not only distinguish WiFi signals from other signals, but also identify signals transmitted by different WiFi APs as each AP may offer different network performance. Fig. 5.4(a) shows the RSS samples taken by a ZigBee radio, which contain signals of 2 WiFi APs and 2 ZigBee nodes.

In Chapter 2 we show the periodicity of 802.11 beacons can be used as a distinctive feature to identify WiFi signals. Specifically, the *folding* algorithm can find the existence of a periodic signal in the original RSS samples. WizNet also applies folding to search for periodic 802.11 beacons in sensor RSS samples. To differentiate individual APs in RSS samples, WizNet uses folding phase as the signature of each AP[52]. Due to the contention-based nature of 802.11 MAC, different APs likely transmit their beacons at different times, resulting in different folding phases. Fig. 5.4(a) shows the RSS samples collected by a ZigBee radio. Fig. 5.4(b) shows the result after folding RSS samples. There are total two peaks in the result. It can be seen that the two peaks have a phase difference, which allows us to distinguish beacons transmitted from different APs.

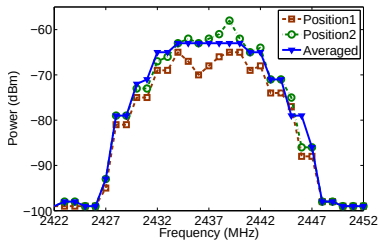
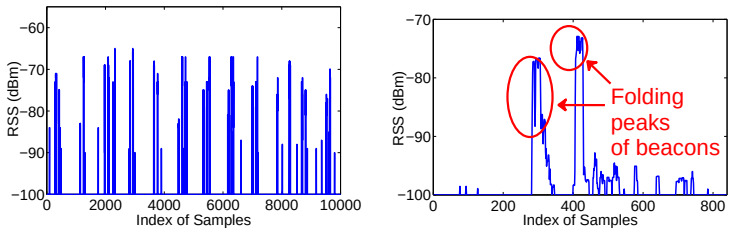


Figure 5.3: Power Density Spectrum of 802.11b signal distorted by multipath fading.



(a) RSS series from ZigBee radio.

(b) RSS after folding.

Figure 5.4: ZigBee RSS samples and folding results. The samples contain signals of 2 WiFi APs and 2 TelosB motes equipped with CC2420 radio.

5.3.3 Sensor Fusion

Hop sampling deals with the frequency-selective fading caused by multipath fading by aggregating the RSS samples collected in multiple ZigBee bandwidths. However, it is only able to handle up

to to 6 dB fading while our experiments show that multipath effect sometimes can vary the RSS for as much as 20 dB at some locations and cause significant spatial variations.

Rician fading [64] is the most commonly adopted stochastic model to characterize the in-door multipath fading. The total power in the dominant paths, denoted as Ω in the Rician distribution, is the RSS value WizNet aims to measure. WizNet adopts a maximum likelihood Ω estimation method proposed by [86]. Specifically, Ω can be estimated by:

$$\hat{\Omega} = \frac{1}{N} \sum_{i=1}^{N} R_i^2 \tag{5.1}$$

where N is the number of samples measured at different locations, and, R_i is the signal amplitude (in mW). We can see from Eqn. 5.1, the maximum likelihood estimate of Ω is essentially the spatial averaging of all the powers of the samples. Fig. 5.3 shows the power density spectrum of an 802.11b node measured by two closely located sensors, and the average RSS computed according to Eqn. 5.1. It can be seen that, the multipath fading can lead to increased (at Position 2) or decreased (at Position 1) signal power depending on the phases of radio waves propagated through different paths, while the sensor fusion effectively mitigates multipath fading by exploiting the spatial diversity of different sensors.

5.3.4 Monitoring AP Scans

User statistics are crucial for WiFi network operators to assess current network usage and plan for future upgrades. The statistics of interest may include the number of users that carry active 802.11 devices and the models of devices in different areas of an enterprise campus. A widely adopted method of collecting user statistics is to log AP data traces. However, this method cannot obtain statistics of potential WiFi users in the areas with little or no WiFi coverage.

A WiFi client discovers APs through either passive or active scanning. Our analysis of various WiFi drivers shows that active scanning is triggered by the actions including: powering on the WiFi NIC, booting up OS, and refreshing the status of available APs. However, 802.11 does not

specify how a client should implement the scanning mode. As a result, different 802.11 drivers may behave significantly different in terms of how the scanning probes are transmitted.

Without being able to decode 802.11 frames, WizNet identifies 802.11 AP scans by searching for the distinctive fingerprints in RSS measurements. Fig. 5.6(a) shows the RSS measurements of AP scanning probes transmitted by two different WiFi clients. One client is an ASUS EeePC netbook running Ubuntu Linux and the other one is Sony TZ27 laptop running Windows Vista. The RSS samples are taken on a TelosB mote listening on 802.11 channel 6. Total 11 peaks can be seen in the RSS measurements of Linux client, which correspond to 11 scanning probes transmitted on 11 different 802.11 channels. Due to channel overlapping and out-of-band emission of 802.11 signals, all probes are captured by the sensor listening on channel 6, although their power magnitudes drop with the increase of channel separation[87]. Although similar phenomenon is observed for the Windows client, a key difference is that two probes instead of one are transmitted on each channel.

Algorithm 2 AP Scan Detection

1: Retrieve an RSS series (denoted as *trace0*) through a sliding window of $1,200$ *ms* from the RSS measurement.
2: Remove samples whose duration does not fall within [366, 732 *us*], and pass the rest through a binary filter, using *TH* as the threshold. The output binary array is *trace1*.
3: Auto-correlate *trace1*, and examine whether it is a periodic signal with a period within [50, 100 *ms*]. If true, store *trace1* as a valid AP scan signature. Repeat from step 1.

We measured the scanning probes of 7 WiFi drivers implemented by 5 different systems: Windows (Vista, XP, 7), Ubuntu Linux 9.1, Symbian 9.3, iOS 4.3.3, and Android 2.2. Our results show that, although the scanning patterns of different WiFi drivers are substantially different, they share the following common characteristics: 1) Probes are sent on all 11 802.11 channels using the same transmission power, although the scanning order might differ; 2) The delay between two probes is constant, resulting in a periodic pattern. Moreover, the period falls within [50, 100 *ms*]. As expected, the period is always shorter than the default 802.11 beacon period (102.4 *ms*) in order to discover APs faster than the passive mode. As a result, the total delay of an active scan procedure lasts shorter than $1,200$ *ms*. 3) The duration of a probe frame is short and typically lasts 366 to

732 *us*. Based on these characteristics, we have developed an algorithm, as shown in Algorithm 1, to identify AP scans from sensor RSS samples.

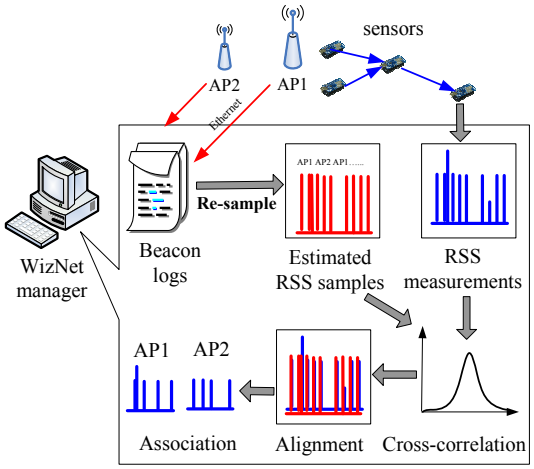
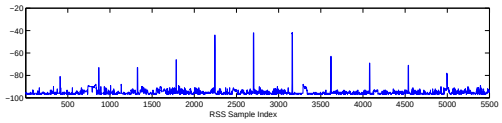
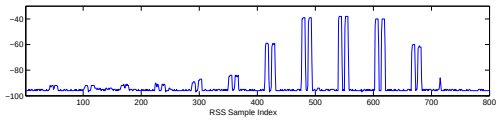


Figure 5.5: The process of RSS and AP association.



(a) ASUS EeePC netbook running Ubuntu Linux



(b) Sony TZ27 laptop running Windows Vista

Figure 5.6: ZigBee RSS measurements of AP scanning probes transmitted by two different WiFi clients.

At step 2, the RSS samples are converted to a binary array by thresholding their magnitude. The auto-correlation operation at step 3 can find the possible period of RSS series, which can accurately identify the model of WiFi client. It requires dot production of RSS samples and incurs high overhead on sensors. In our implementation, the binary RSS array output at step 2 is compressed and transmitted to the sink, which then executes the auto-correlation.

5.4 Design of WizNet Manager

5.4.1 RSS and AP Association

As discussed in Section 5.3.2, WizNet can distinguish the beacon frames sent by different APs based on the folding phases of RSS samples. However, it still cannot obtain the identities of the APs associated with the beacon frames. The correct association of each AP with the RSS samples it generated is essential to keep track the performance of individual APs. A straightforward solution is to compare the timestamps of RSS samples recorded by WizNet sensors and those of 802.11

beacons recorded by APs. However, this approach would require millisecond-level precision of time synchronization between different APs and sensors, which incurs high overhead.

To associate APs with their RSS samples, WizNet applies a signal processing technique called *cross-correlation* which does not require high-accuracy time synchronization between APs and sensors. The basic idea is illustrated in Fig.5.5. First, each AP periodically logs beacon frame headers and sends to WizNet manager. The WizNet manager then merges all AP logs into a single log and converts it to an RSS time series through a *re-sampling* process in which an RSS series is generated based on the timestamps and data rates contained in beacon headers. This process essentially simulates the sampling operation of sensors based on the native time of AP. The RSS magnitude is set to either a fixed value when the sampling time falls within a beacon duration or zero otherwise. Both the generated RSS series and the folded RSS series from a sensor are then fed into a *cross-correlator* that computes the dot product of two series with different offsets. The maximum dot production corresponds to the most likely alignment offset between the two series. This offset is essentially the error between the system times of AP and sensor. Finally, the sensor RSS samples are shifted according to the offset found and then labeled by the BSSID of the matched AP.

We note that WizNet is not a standalone system due to the need of AP logs. However, AP logs are typically easy to obtain on most off-the-shelf production APs. For example, many APs run Linux systems that provide various tools for extracting system logs. Moreover, as AP logs are usually very short (several KB per second), collecting them does not incur much overhead over the network infrastructure. If the system needs to be strictly standalone, WizNet can also obtain AP logs from dedicated 802.11 sniffers which capture beacons from nearby APs.

5.4.2 SNR and Channel Utilization Estimation

After associating RSS samples with APs, WizNet manager first calculates the sensor SNR by subtracting the base noise of the ZigBee sensors from the RSS samples. The base noise is computed by applying exponential moving average over the minimum values in the RSS series. Then the

manager infers the SNR of APs that a WiFi client would receive at every monitoring spot.

Suppose a WiFi client and a WizNet cluster are located at the same location. The signal strength of an 802.11 frame is w_w and w_z at the virtual antennas of WiFi and WizNet receivers, respectively. As WizNet fuses the RSS from multiple sensors, the virtual antenna comprises all the antenna of sensors in a cluster. G_w and G_z are the virtual antenna gains of WiFi and WizNet receivers, respectively. Then the signal to noise ratios of the receivers, denoted as SNR_w and SNR_z, can be expressed as:

$$SNR_w = 10log(w_w G_w p_w / N_w) \tag{5.2}$$

$$SNR_z = 10log(w_z G_z p_z / N_z) \tag{5.3}$$

where p_w and p_z are the ratio between the signal power measured by receiver RSSI and the total signal power, N_w and N_z are the receiver noise floors. The sensor fusion of WizNet mitigates the multipath fading in each cluster. As a result, if the virtual antennas of WiFi and ZigBee are sufficiently close, $w_w \approx w_z$. G_w and G_z are functions of the incoming signal bearing which can be considered same for closely located virtual receivers. Subtracting Eqn. 5.3 from Eqn. 5.2 gives the difference between SNR_w and SNR_z:

$$SNR_w = SNR_z + C \tag{5.4}$$

where C can be estimated from a simple and short training phase. A different model is estimated for each monitored AP because C is a function of signal bearing which varies with the location of source.

A key advantage of our SNR estimation approach is that the model training usually only needs to be performed once before deployment, because of two reasons. First, the noise, multipath and frequency-selective fading that can significantly affect the mapping between ZigBee and WiFi SNRs are effectively dealt with at run time by hop sampling, folding and sensor fusion of WizNet. Second, the SNR mapping model in Eqn. 5.4 only characterizes the difference between ZigBee and WiFi measurements *after* the impact of these dynamics is accounted for. As a result, this model

does not need to be retrained frequently at run time. Our experiments that last more than six days show that WizNet can achieve satisfactory monitoring fidelity after a single offline training phase (see Section 5.5).

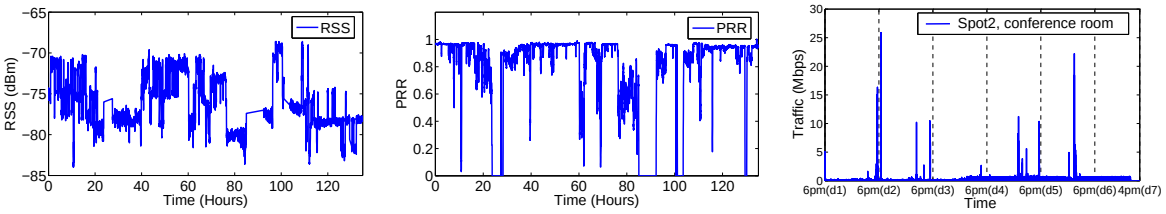
WizNet manager then estimates the utilization rates of the APs' working channels. Due to the sharing nature of wireless channels, only non-occupied time slots on a channel can be utilized by clients. WizNet manager computes the channel utilization rate as the ratio between the number of RSS samples whose signal strengths are above the noise threshold, and the total number of RSS samples.

5.4.3 Throughput Estimation and Rogue AP Detection

The SNR and channel utilization rate can be used to infer the throughput of WiFi. We now outline the basic idea and leave the detailed design for future work. We build the empirical model in an offline training phase in which the throughput of a reference 802.11 client is measured under different SNR and channel utilization rate combinations. Due to the rate adaptation of WiFi receivers, the throughput experienced by the reference client is a range of values. WizNet manager then estimates the current throughput by searching the best match of the measured {SNR, Utilization Rate} pair in the training data set.

WizNet is able to discover rogue APs that are deployed without the authorization of network administrators. Rogue APs may lead to security breach since they can be exploited by third parties to access the secured networks. As discussed in Section 5.4.1, the RSS measured by sensors are associated with the APs using cross-correlation between the two RSS traces obtained by sensors and APs, respectively. WizNet manager labels each identified RSS in the series obtained by sensors with BSSIDs of the APs. As a result, any AP that cannot be identified is potentially a rogue AP. In this approach, each AP is identified by the temporal phases of its beacon transmissions. By leveraging such PHY layer information, WizNet can reliably detect rogue APs even if they can forge their SSIDs and MAC addresses.

5.5 Experimentation



(a) RSS evolution over time. (b) PRR evolution over time. (c) Traffic evolution over time.

Figure 5.7: The network dynamics observed in a large conference room.

5.5.1 System Deployment and Experimental Settings

We have implemented the sensor components of WizNet in TinyOS 2.x on Crossbow TelosB motes. The sensor code has a footprint of 16 KB and uses 550 bytes of RAM. We are able to achieve a sampling rate as high as 32.768 kHz on TelosB motes. However we deliberately decreased the rate to 8.192 kHz to conserve energy. The WizNet manager is written in C and Python. We implemented a single clustering protocol that can synchronize the sensors in a cluster and allow the cluster head to communicate with the base station. WizNet manager learns the working channels of monitored APs from AP logs, and instructs the WizNet sensors to sample these channels.

We deploy WizNet on the 3rd floor of the engineering building of Michigan State University. A production 802.11b/g/n WiFi network containing 115 APs is currently available in the building. Over 50 physical APs can be detected on each floor, which are almost evenly distributed on channel 1, 6 and 11. Our deployment consists of 20 TelosB sensors and 6 802.11 laptops, as well as one desktop computer as manager. We divide the 20 sensors into 5 clusters, and deploy them to five rooms on the third floor. To compare the monitoring accuracy of WizNet against 802.11 based tools, we also collect measurement results from 802.11 laptops which sniff WiFi traffic. These laptops are deployed close to WizNet clusters. To account for sufficient environmental diversities, we deploy cluster 1 and 3 in two small offices, cluster 2 and cluster 5 in two conference rooms, and cluster 4 in a medium size mail room. The small offices have very few people traffic, while the two conference rooms are frequently occupied by meetings and seminars. The mail room not only

has many people visiting, but also has a few fixed tall metal cabinets which substantially block signals. Our deployment covers a floor area of approximately 46,000 square feet that is serviced by 11 production APs, as shown in Fig 5.8.

Our experiment lasts for a period of 140 hours, during which 40 GB data is collected. As we cannot deploy code to the monitored production WiFi network, we extracted AP logs (required by the AP association component) from the traces logged by our own laptops. In order to capture fine-grained WiFi performance variability, the sensors measure and report the data to sink every 10 seconds. During this period, each sensor keeps active for 2 seconds. Our measurement shows that the TelosB motes in our deployment consume 22 mA and 10 uA in active and sleep states, respectively. If the motes are powered by 2 2500 mAh AA batteries, the system can last over 150 days with 3% duty cycle (i.e., the monitoring data is reported to the manager every one minute). An initial training process is conducted to train the SNR estimator of each cluster using the measurements from both WiFi client and sensors. After training, the absolute estimation error is computed between the SNR estimated by WizNet and the ground truth SNR measured by the WiFi client at each monitoring spot.

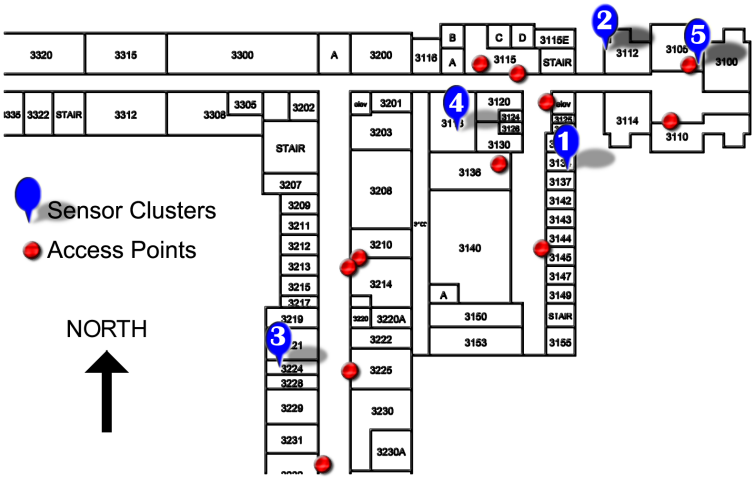
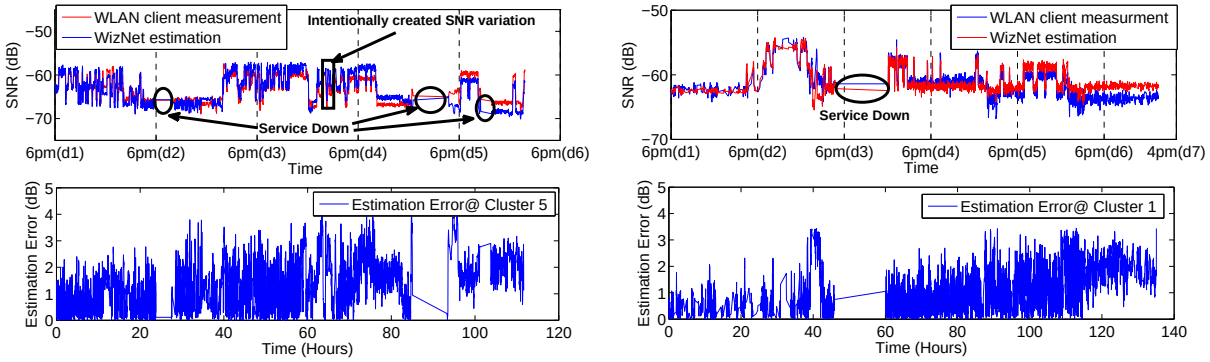


Figure 5.8: Locations of production WiFi APs and monitoring WizNet clusters on the third-floor of engineering building of a university. The total deployment area is about 46,000 square feet.

5.5.2 Network Dynamics

Over 25 APs were observed during the period of 140 hours. However, some distant APs have consistently weak signals throughout our experiment and clients normally do not associate with these APs. We focus on the results of 11 APs shown on Fig 5.8. The 140-hour trace logged by WiFi clients shows that the network yields significant dynamics. Fig 5.7 (a) and (b) also show strong correlation between the SNR and the PRR of 802.11 beacons. This indicates that SNR is a good metric to evaluate WiFi performance. We notice that there were several service breakdowns during which the signals from some APs were not observed. Moreover, the network usage also fluctuates significantly during the experiment. Fig 5.7 (c) shows that the aggregated traffic rates on all channels at one conference room vary between near zero to 25 Mbps, while the mean traffic rate is only 36 Kbps. The traffic yields large bursts, which is occurred during normal office hours. WizNet observed 4 APs that cannot be identified by the beacon logs from the monitored 802.11 network. After a careful analysis of the data traces logged by laptops, we found these APs are not a part of the production network hence they are considered as rogue APs.

5.5.3 Monitoring Accuracy



(a) Cluster 5 estimation error vs time.

(b) Cluster 1 estimation error vs time.

Figure 5.9: Estimation error vs time.

We have extensively evaluated each component of WizNet. The results are summarized here while the details are omitted and can be found in a technical report [98]. (1) Hop sampling is able

to reduce the RSS estimation variation by about 5 dB, and sensor fusion can further reduce the variation by another 4 to 6 dB. (2) Fusion of two sensors can already substantially improve the SNR estimation accuracy by exploiting the higher degree of spatial diversity, while the benefit of fusing more than three sensors is insignificant.

We now evaluate the impact of training time on the accuracy of SNR estimator. The absolute estimation error is computed between the SNR estimated by WizNet and the real SNR measured by the laptop. Each CDF in Fig. 5.12 includes the errors of all four APs monitored at spot 2. We can see that the error decreases when a longer training period is used. However, even when the system is only trained for 200 seconds, 90% of errors over the period of 140 hours fall below 3 dB. When the training time is prolonged to 500 seconds, only slight performance gain is achieved. We adopted a training period of 200 seconds in the following experiments.

Next we evaluate the impact of sensor fusion on estimation accuracy at monitoring spot 2 by varying the number of sensors. Fig. 5.10 shows that the absolute estimation errors of all 4 monitored APs become smaller when the number of sensor increases. When the number of sensors is sufficiently large (> 3), the error decrease becomes insignificant. Fig 5.11 shows the RSS estimation of all clusters over the 140-hour period. We can see that clusters 2 and 4 perform slightly worse than other clusters. This is because they are placed in the two conference rooms where passing pedestrians are constantly present during normal work hours. Nevertheless, 80% of errors fall below 2.5 dB.

5.5.4 Spatiotemporal Performance Analysis

The results in the previous section show that WizNet yields satisfactory monitoring accuracy during the 140-hour evaluation period. We now analyze the micro-scale spatiotemporal performance of the system. We focus on the analysis on 2 APs that experienced the highest dynamics in our experiment. Fig 5.9 shows the ground truth and estimated SNR of the two APs (monitored by cluster 1 and cluster 5, respectively). It can be seen that both APs yield significant performance variability. However, both clusters are able to accurately track the dynamics of the APs and maintain SNR es-

timation errors within 4 dB. This result indicates that the SNR variation caused by environmental factors has little impact on WizNet monitoring fidelity. Instead, we observe that the estimation errors gradually increased over time after the initial training. We suspect that the increased inaccuracy is largely attributed to the radio hardware drifts caused by temperature and humidity changes. However, the overall error increase is within 1 dB during the period of 6 days. As shown in Fig 5.12, a longer training length (2 to 3 minutes) would give more consistent estimation accuracy.

We also notice that, once WizNet is properly trained, its performance is resilient to dynamic obstacles in the environment. This is confirmed partially by the fact that people traffic is regularly present in our testing areas. Moreover, we deliberately rearranged some furniture, including chairs and tall metal shelves near cluster 5 after training during the experiment, and the time period is marked on 5.9 (a) by a rectangle. Although substantial variation was observed from SNR measurements, WizNet still maintains a small estimation error compared with the measurement of 802.11 laptop. This is due to the fact that the hop sampling and sensor fusion components effectively mitigated the dynamics of multipath fading in the environment.

5.5.5 Client Classification

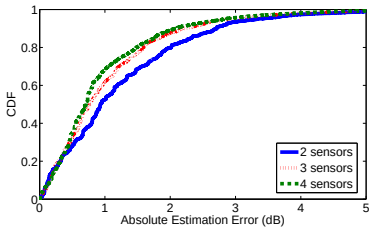


Figure 5.10: CDF of estimation errors vs sensor number.

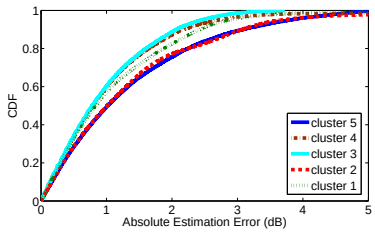


Figure 5.11: Error of RSS estimation of all clusters.

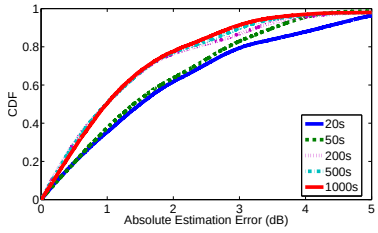


Figure 5.12: CDF of estimation errors vs training length.

We evaluate the accuracy of monitoring AP scans using a WizNet senor placed in an office without WiFi coverage[1]. Users carrying different 802.11 client devices listed in Table 5.1 roam about the testing area. In the first experiment, only one user appears in the area at a time. The

[1]WizNet obtains the user statistics directly from AP logs in the areas with WiFi coverage

Table 5.1: False Positive and Negative Rates of 802.11 Client Classification.

Client	FN Rate (single-/multi-client)	FP Rate
Nokia E52-Symbian 9.3	6% / 9%	3% / 5%
Apple iPhone-iOS 4.3.3	6% / 11%	8% / 15%
Lenovo X200-Win 7	10% / 19%	2% / 2%
Sony S26C-Win XP	2% / 4%	7% / 13%
ASUS EeePC-Ubuntu 9.1	2%	10%
HTC Desire-Android 2.2	5%	5%

results evaluate the performance of AP scan recognition algorithm presented in Section 5.3.4. Each client device performs 100 active scans. A Ubuntu Linux laptop is used to record the sniffed AP scans as ground truth. The WizNet sensor starts with no knowledge of any scan patterns. The overall accuracy of the system is shown in Table 5.1 (columns labeled as “single-client"). It can be seen that the classification accuracy varies for different clients, due to the fact that AP scans of some systems have more evident features than others. However, all the classification errors fall below 10%. In the second experiment, four users carrying different client devices appear in the testing area at the same time, and each client issues 100 scans. Table 5.1 (columns labeled as “multi-client") shows that the AP scan probes transmitted by 802.11 driver of Windows 7 are substantially shorter than other systems, making them easier to be missed in RSS sampling. Moreover, the features of iOS AP scans are less distinctive. As a result, 15% of these scans are mistakenly classified as from other systems.

CHAPTER 6

CLOCK SYNCHRONIZATION IN WIRELESS SENSOR NETWORKS

6.1 Introduction

Time synchronization is a fundamental service for Wireless Sensor Networks (WSNs). Many applications of WSNs require the nodes to maintain a common notion of time. Samples from different nodes often need to be temporally correlated in order to infer the information of interest. Accurate and precise timestamping is thus essential for correct data ordering and processing. Moreover, a common time representation is critical for the energy efficiency of battery-powered nodes that must coordinate their sleep schedules and communication activities.

We propose a new time synchronization approach for WSNs, which exploits the existing WiFi infrastructure. Our approach is motivated by two recent trends in wireless technologies. First, WiFi networks have enjoyed a phenomenal penetration rate in our society in the past decade. One of the key reasons attributed to the popularity of WiFi is the adoption of unlicensed radio spectrum, which enables the proliferation of inexpensive off-the-shelf 802.11 devices. Second, recent WSN platforms have embraced low-power wireless standards such as Bluetooth and 802.15.4, which also adopt the 2.4 GHz unlicensed spectrum. As a result, WSNs and WiFi networks often occupy same radio frequency bands. Such wireless co-existence is often deemed as a “curse" as it may cause significant interference between different platforms [55].

In this chapter, we exploit the co-existence of WiFi and 802.15.4-based WSNs as a “blessing". The 802.11 standards require all WiFi access points (APs) to broadcast periodic *beacon* frames for the purpose of network management. Working on the same radio frequencies, 802.15.4 sensors can detect the transmissions of such beacons and use them as a clock signal to synchronize their clocks. This approach has several key advantages. First, it does not require any modifications to 802.11 APs, and thus can leverage the ubiquitous WiFi deployments. Second, our measurements

show that many production WiFi APs have a communication range of hundreds of feet even in complex indoor environments, which is about an order of magnitude longer than that of 802.15.4 nodes. In addition, the distribution of 802.15.4 networks are typically "denser" than AP. As a result, a large number of sensors within a connected WSN cluster can synchronize to the same beacons to achieve network-wide global time with very infrequent message exchanges. Despite the aforementioned advantages, several challenges need to addressed before this approach becomes viable in practice. First, as 802.15.4 radio cannot decode any 802.11 frames, they must rely on sensing the Radio Signal Strength (RSS) of in-air signals to identify 802.11 beacons. This is not trivial as RSS samples also contain non-beacon signals such as 802.11 data frames and transmissions of other 2.4 GHz devices such as cordless phones. Second, since 802.11 adopts a CSMA (Carrier Sense Multiple Access) MAC, a beacon transmission may be delayed due to channel contention, making it challenging to achieve sub-millisecond synchronization accuracy required by many WSN applications.

This chapter makes the following contributions. First, we conduct a large-scale measurement study of 802.11 beacons in an enterprise WiFi network consisting of over 50 APs deployed in a 300,000 square foot office building. We experimentally characterize the spatial coverage of WiFi APs and the temporal characteristics of beacons. We show that the periodicity of 802.11 beacons is highly stable despite the existence of small jitters caused by heavy data traffic.

Second, we implement WizSync in TinyOS 2.1.1 and conduct extensive evaluation on a testbed consisting of 19 TelosB motes. Our results show that WizSync can achieve an average synchronization error of 0.12 milliseconds over a period of 10 days with power consumption of 50.9 microwatts/node.

6.2 Problem Statement

6.2.1 Background on 802.11 Beacons

802.11 requires all APs to broadcast periodic *beacon* frames that carry important management information (e.g., supported rates and security settings). The default beacon period is 102.4 *ms*, which is rarely changed on production APs. Since 802.11 adopts CSMA (Carrier Sense Multiple Access), the transmission of beacon frame may be delayed due to channel contention caused by pending or ongoing data transmissions. However, as defined in 802.11, whether a beacon frame is delayed or not, the subsequent beacon frame shall always be scheduled at the undelayed nominal beacon interval.

6.2.2 Clock Synchronization via 802.11 Beacons

Many WSN platforms generate on-board clock signal from low-power CMOS crystal oscillators, which often suffer significant drifts. The crystal oscillator of TelosB mote has a drift rate of $30-50$ *ppm* [74]. As a result, the clocks of nodes need to be frequently calibrated and synchronized to achieve accurate timekeeping across the network. In this chapter, we propose a new approach that employs the periodic 802.11 beacons broadcasted by WiFi APs as a global timekeeping signal to synchronize the clocks of 802.15.4-based sensor nodes. In contrast to other external clock based approaches, our approach requires no additional hardware as off-the-shelf 802.15.4 radio is capable of sensing 802.11 transmissions.

Our approach is designed to meet two objectives: 1) *Accuracy.* Specifically, clock synchronization errors across different nodes should not exceed 1 *ms*. Such an accuracy requirement can satisfy the need of timekeeping in many WSN applications. Several WSN time synchronization systems such as Syntonistor [79] are also designed to achieve similar level of accuracy. 2) *Energy-efficiency.* Due to the tight energy budget, nodes should minimize radio transmission and idle time during clock synchronization.

6.3 A Measurement Study of 802.11 Beacon

We conducted two experiments to study the spatial and temporal characteristics of 802.11 beacons in an enterprise production WiFi network deployed in the engineering building of Michigan State University. The building is a four-story complex with a floor area of approximately 300,000 square feet. A production WiFi network containing 115 APs is currently available in the building. Over 50 physical APs can be detected on each floor, which are almost evenly distributed on channel 1, 6 and 11. These APs are manufactured by several different vendors.

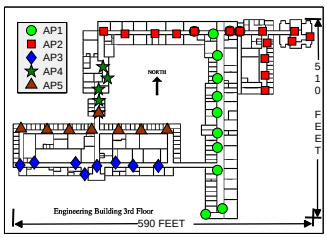
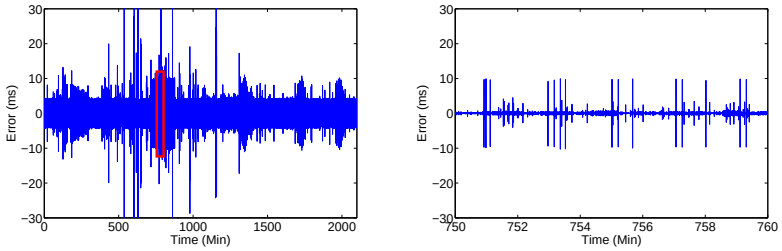


Figure 6.1: The coverage of 5 APs on the third floor of Engineering Building at Michigan State University.



(a) Jitters of beacon periods vs. time.

(b) Beacon period jitter vs. time. The results are excerpted from the rectangle in (a).

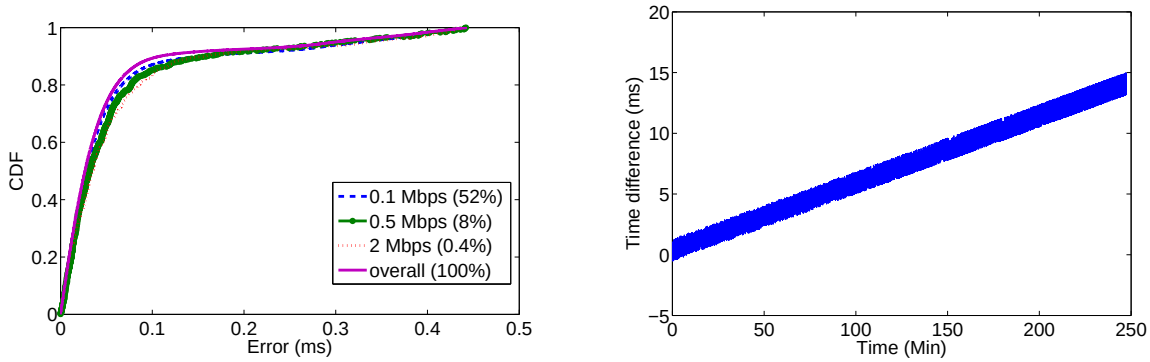
Figure 6.2: Temporal stability of 802.11 beacon period.

6.3.1 Spatial Coverage of 802.11 APs

In the first experiment, we carried out a WiFi coverage site survey on the third floor of the building. A laptop is carried by a user who roamed about the corridors on the third floor. The user stops for 10 seconds for every 10 seconds he walked. When he stops, the carried laptop begins to passively scan for nearby APs on channel 1, 6 and 11. For clarify of presentation, we only shows the coverage map of 5 APs on channel 11 in Fig. 6.1. It can be seen that each of these APs covers a very large area. For example, AP1 fully covers the vertical corridor which is about 510 feet long. This distance is about an order of magnitude longer than the typical indoor communication range of 802.15.4 radio. This result clearly demonstrates a key advantage of WizSync, i.e., nodes distributed in a large region may synchronize their clocks without any message exchange.

6.3.2 Temporal Stability of 802.11 Beacon Period

In the second experiment, we deployed four laptops at different locations of the building to measure the timing accuracy of beacon transmissions. During a period of 2 days, the four laptops record all overheard beacon frames, reception timestamps, signal strength, etc. To evaluate the impact of traffic load, the laptops also record the headers of 802.11 data packets and periodically compute the average traffic rate. We measure the interval between the reception timestamps of two consecutive beacons of the same AP. The relative errors between the measurements and the standard beacon interval (102.4 *ms*) are then computed.



(a) CDF of beacon period jitters. (b) Time difference bw 2 APs.

Figure 6.3: The temporal characteristics of 802.11 beacons.

Fig. 6.2(a) shows the beacon period jitters during the experiment of 2,000 minutes. It can be seen that most of the jitters fall below 5 *ms*. However, we observed that only a small number of beacons yield jitters higher than 1 *ms*. Fig. 6.2(b) shows the “zoomed in" results taken from the rectangle of Fig. 6.2(a), which contains measurements of 600 seconds. We can see that most of the jitters are nearly zero and high jitter only appears occasionally. This observation implies that a simple outlier removal process can effectively eliminate the high jitter. After removing 15% outliers, the resulted maximum jitter is only 200 *us*.

An interesting observation from Fig. 6.2(b) is that high jitter typically occurs in burst. Our analysis of the data trace indicates that such bursts were caused by heavy traffic on the channel. Fig. 6.3(a) shows the CDF of beacon period jitters (10% outlier removed) under different channel traffic loads. It can be seen that the error increases with the traffic load, which is consistent with

the expectation. Nevertheless, even when the traffic load is as high as 2 *Mbps*, 90% of the beacons are transmitted with an error smaller than 200 *us*. The average beacon period jitter during the 2000-minute experiment is 80 *us*. We also note that the high traffic load is rare in the network. For instance, 2 *Mbps* is reached only in 0.4% of the time. This observation is consistent with the results several empirical studies based on production WiFi deployments [49]. We also measure the difference between the timestamps carried by the beacons sent from different APs, which indicates the time synchronization errors of APs. We observed that the clocks of different APs are not synchronized. Fig. 6.3(b) shows the time difference between two APs over a period of 250 minutes. The difference increases linearly over time, which conforms to the results in Fig. 6.3(a) that the frequency of each clock is highly stable. This result suggests that, when two sensors are synchronized to different APs, their clocks will yield a linear frequency difference. We will leave the synchronization in multiple AP scenario as our future work.

6.4 Experiment

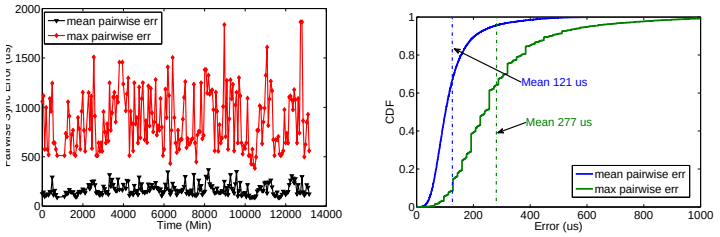
We designed a novel time synchronization protocol called WizSync, which employs advanced signal processing techniques to detect periodic WiFi beacons and use them to calibrate the frequency of native clocks. The details of our design can be found in [52]. We implement WizSync in TinyOS 2.1x on the TelosB platform. Our implementation has a code size of 4 *Kbytes* and memory usage of 1 *Kbytes*. We conducted a 10-day experiment on our testbed consisting of 22 nodes to extensively evaluate the long-term performance of WizSync. In this section, we first present our experimental methodology, followed by the experiment results and the in-depth analysis.

6.4.1 Experimental methodology

Our testbed is composed of 3 WizSync node clusters deployed in the Engineering building. Three rooms on the third floor are selected to host the nodes. These rooms lie on a straight line of 300 feet with roughly equal spacing. Located at different sections of the building, they are separated

by many offices and labs between them. As a result, no single AP can cover any two of these rooms. Each cluster consists of one laptop and several TelosB motes running WizSync which are connected to the laptop with USB cables. The positions of nodes are randomly chosen within the room. The laptop collects the data from the WizSync nodes, as well as logs the overheard 802.11 frame headers, which provides traffic information for analysis. We manually choose a node as cluster head in each cluster.

During initialization, the cluster head finds the AP with the strongest signal and notify other nodes in a broadcast message, as discussed in Section V.C. The broadcast message also sets the same initial clock values of all nodes in the cluster. In order to compare the local times of cluster members, the cluster head broadcasts a message every 30 seconds after initialization. The cluster members report their local times to the laptop when the packet is received. Our evaluation last 10 days continuously during which over 40 GBytes data is gathered from 19 WizSync nodes and 3 laptops.



(a) Synchronization error over time.

(b) CDF of synchronization errors.

Figure 6.4: Intra-cluster synchronization errors of 19 nodes in a period of 10 days.

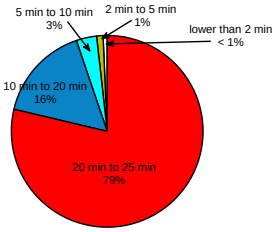
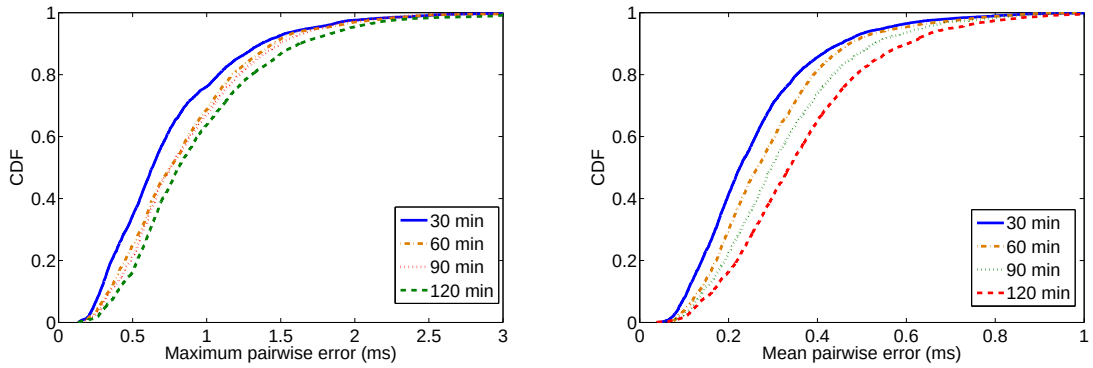


Figure 6.5: Sleeping time distribution of 19 nodes in a period of 10 days.

6.4.2 Intra-cluster synchronization

We first analyze the synchronization errors between the nodes within the same cluster. Fig.6.4(a) depicts the mean and maximum pairwise errors of all 19 WizSync nodes in a period of 14,000 minutes. The pairwise error between any two nodes is computed every 30 seconds. Each data point in Fig. 6.4(a) is the *maximum* value of the mean or maximum pairwise errors of all node pairs in a 30-minute window. It can be seen that the mean error falls below 400 *us* in most of the time while



(a) Mean pairwise errors. (b) Maximum pairwise errors.

Figure 6.6: CDFs of inter-cluster synchronization errors of 10 nodes over a period of 7000 minutes.

the maximum error typically ranges from 500 to 1,300 *us*. Moreover, although the average error remains stable over time, the maximum error yields significant fluctuations. Our analysis indicates that most of the spikes are the result of beacon backoffs caused by sporadic heavy traffic. Fig. 6.4(b) shows the CDF of mean and maximum errors of all the experimental data. It can be seen that 90% of the mean and maximum error are below 200 *us* and 450 *us*, respectively. The mean values are 121 *us* and 277 *us*, respectively. Our results indicate that WizSync not only achieves high synchronization accuracy but also has low power consumption. Fig.6.5 shows the sleep time distribution of all 19 nodes during the 10-day experiment. Nodes remain awake longer than 20 minutes in 79% of the total experiment time. Short sleep intervals (less than 2 minutes) only occurred in less than 1% of the total experiment time. In such a case, sensor clocks experience significant transient drifts and WizSync have to wake up nodes frequently for calibration. The average power consumption of a WizSync node during the 10-day experiment is 50.9 *uW*.

6.4.3 Inter-cluster synchronization

We also evaluated the accuracy of WizSync across different clusters that synchronize to different APs. However, different sensor clusters in our deployment cannot communicate with each other due to the long distance between them. We conducted a trace-driven simulation as follows. We implemented the calibration and offset correction algorithms of WizSync in Matlab and fed them

with data traces collected from two 5-node clusters over a continuous period of 7,000 minutes. We adjust the interval of message exchanges for offset correction and measure the resulted synchronization errors.

Fig.6.6(a) and Fig.6.6(b) show the mean and maximum synchronization errors, respectively. As expected, the errors increase with the duration of message exchange interval. However, even with the interval of 120 minutes, 90% of the maximum and mean pairwise errors still fall below 1,500 *us* and 600 *us*, respectively. When the interval is 30 minutes, the average value of the maximum and mean pairwise errors are 250 *us* and 754 *us*, respectively, which are well below the preset error 1 *ms* error bound.

CHAPTER 7

PRESERVING NFC PHYSICAL SECURITY

7.1 Introduction

In recent years, the Near Field Communication (NFC) technology is increasingly available on the new generation of smartphones, tablets, and smart accessories. It is estimated that more than 200 million NFC-enabled smartphones will be shipped in 2013 [16]. And over 50% of the smart devices to be shipped in 2015 will have NFC support [11]. The growing popularity of NFC has enabled a range of applications, from contactless payment [15] and ticketing [28] to device pairing [27] for ad hoc data exchange.

A major trait of NFC is its short communication range (usually within 10 cm), which is the result of the fast decaying magnetic induction between the antennas of NFC transmitter and receiver. The short communication range is favored by many security-sensitive applications, such as contactless payment, since it provides a natural, physical protection against various attacks, particularly malicious eavesdropping. Unfortunately, as NFC is still a relatively new and developing technology, its implementation on mobile devices often have design flaws, which may be exploited to compromise application security [68]. In particular, our experimental study described in this work shows that, current NFC radios emit significantly more RF energy than intended. With a specially designed portable NFC sniffer, we are able to eavesdrop NFC transmissions from up to 240 cm away, which is at least an order of magnitude further than the intended NFC communication distance. These findings raise major concerns on the physical security of NFC. Moreover, this issue is aggravated by the fact that current NFC chipsets adopt fixed transmission power, which cannot be adjusted to mitigate the potential risks of eavesdropping.

Existing efforts on NFC security can be classified into two basic categories. Several solutions improve the security of NFC by adding more security elements, such as additional secret keys, to

the native OS of mobile devices [53]. However, the mobile device would become vulnerable if the integrity of the OS is compromised (e.g., after being rooted). The second category employs additional hardware devices to secure NFC [76][22]. However, these hardware systems are bulky and power-hungry, which are ill-suited for mobile devices. In a recent work [50], a hardware security device is developed to harvest energy from NFC transmissions and jam malicious interactions. However, due to the low energy harvesting efficiency, the system may not provide uninterrupted protection. The above approaches are designed to prevent content-based malicious attacks, and none of them can protect NFC from eavesdropping attacks.

In this chapter, we propose a novel, noninvasive NFC security system called *nShield* to protect NFC against passive eavesdropping. nShield is a credit card-sized thin pad that can be easily stuck on the back of mobile devices (see Fig. 7.6). nShield implements a novel adaptive RF attenuation scheme, in which the extra RF energy of NFC transmissions is determined and absorbed by nShield. At the same time, nShield scavenges the extra RF energy to sustain the perpetual operation. A key contribution of this work is the analysis of the factors affecting the energy harvesting efficiency, and the design of a highly effective energy harvesting system. nSheild is capable of harvesting significant amount of power (55 mW) from commodity mobile devices, which is at least a 1.8X improvement over the state-of-the-art NFC-based energy harvesting systems. Together with the extremely lo-power design, it enables nShield to provide the host uninterrupted protection against malicious eavesdropping. Lastly, the small form factor, self-sustainability, and transparency to OS, makes nShield an attractive solution to retrofit existing mobile devices with protection against passive eavesdropping.

In summary, we make the following key contributions in this chapter.

1. We conduct an experimental study on the feasibility of passive NFC eavesdropping, with a specially designed inexpensive NFC sniffer. We show that commodity NFC-enabled devices can be eavesdropped from up to 240 cm away, which is at least an order of magnitude further than the intended NFC communication distance. Moreover, although external signal attenuation is effective in reducing NFC transmission power, the desired attenuation level

that can still sustain data communication is highly dependent on the NFC hardware, tags sensitivity, and the physical distance. To our best knowledge, this is the first empirical study on passive NFC eavesdropping in practical settings.

2. We design an NFC security system called nShield to protect NFC from passive eavesdropping attacks. As a key novelty, nShield absorbs the excessive RF energy of NFC to attenuate the signal strength against passive eavesdropping, while the absorbed RF energy is scavenged for its perpetual operation. By exploiting the NFC target discovery process, nShield intelligently determines the right attenuation level that is just enough to sustain reliable data communication. As a result, it can promptly and precisely control the signal strength of NFC transmissions, mitigating the risk of passive eavesdropping.

3. We carefully analyze the factors that affect the NFC energy harvesting efficiency, and apply several design techniques to the antenna and hardware of nShield to maximize the amount of harvested energy, which include quality factor optimization, voltage matching, and tag emulation. As a result, nShield can harvest significantly more power (1.8X and 3.1X) than the two state-of-the-art NFC energy harvesting systems. This capability enables nShield to provide the host uninterrupted protections against passive eavesdropping attacks.

4. We implement a prototype of nShield, and evaluate its performance via extensive experiments. Our results show that nShield has extremely low power consumption, high energy harvesting efficiency, and can adaptively attenuate the signal strength of NFC transmissions in a variety of realistic settings, while only introducing insignificant delay.

7.2 Background

NFC employs the fast decaying magnetic induction between the antennas of transmitter and receiver for communication in close distance. The typical working distance of NFC using compact antenna coils (with the size of a credit card) is a few centimeters. An NFC communication process

involves an initiator and a target. Initiator devices are usually smartphones, tablets, and POS terminals, which initiate the NFC communication with the target. The target devices can either be those devices or proximity cards. NFC has two working modes, i.e., passive mode and active mode. The passive mode employs the same communication techniques as those used by the proximity card, in which the target device is powered by the RF field emitted by the initiator, and transmits by modulating the RF field. In the active mode, both initiator and target are powered by their own energy sources. The ASK and PSK modulation schemes are employed by NFC to support a number of data rates (106 kpbs, 212 kbps and 424 kbps).

An NFC communication process always begins with *target discovery*, in which the NFC initiator discovers the nearby NFC targets and learns the capability of the discovered targets. The initial phase of discovery process is probing, in which the initiator broadcasts discovery messages periodically to find nearby target devices. An NFC target device responds after it hears the probe. The initiator and the target then exchange a few parameters back and forth to learn the capabilities of each other before the start of the real data communication. On an NFC-enabled Android phone, when the screen of the phone is unlocked, the NFC radio is activated and the discovery process starts automatically and continues until a target device is discovered. During this process, the discovery probes are broadcast at a frequency of about 1.4 Hz. Using NFC antennas, a device can harvest energy from the RF field generated by NFC initiators within close proximity (a few centimeters). However, the amount of energy that can be harvested during the probing is usually very limited, as NFC radios have a low duty-cycle (10%) during the probing phase.

Passive eavesdropping attacks are harmful to wireless communications in several ways. They could not only compromise the privacy/security of the system, but also serve as the early steps of other more damaging attacks [89], e.g., the man-in-the-middle attacks [89]. Another reason that makes passive eavesdropping attacks especially harmful is that they are hard to detect, as they do not actively transmit any signal and are usually launched from distance. NFC is generally considered to be a secure wireless technology against eavesdropping, due to its short communication range. However, current NFC implementations often emit significantly more RF power than

intended. Our study shows that, with specially designed NFC sniffers, NFC signals can be eavesdropped from as far as 2.4 m away, which is much further than the intended NFC working distance. This poses a serious concern for security/privacy-sensitive NFC applications such as contactless payment.

7.3 A Measurement Study

In this section we experimentally study the passive eavesdropping distance of NFC transmissions. Specifically, we measure the physical distance at which the signals from initiators and targets can be successfully decoded, i.e., eavesdropped. Moreover, we study the impact of transmission power attenuation on the passive eavesdropping distance of different NFC devices. The results provide important motivation for the design of nShield.

We note that the actual eavesdropping distance depends on many factors, such as initiator implementation, initiator position, NFC working mode (active or passive), and environmental factors (e.g., background noise). Our measurements are conducted in typical settings, and an exhaustive evaluation of all these factors is beyond the scope of this chapter. Nevertheless, our results raise serious concerns about the physical security of NFC due to the significant discrepancies between the actual and intended working distances, and shed lights on possible defense mechanisms.

7.3.1 Experimental Setup

Our experiment is conducted using NFC initiators, tags, and a sniffer. Commercial off-the-shelf NFC transceivers do not make good sniffers for two reasons. First, they typically have a small antenna size due to the form factor constraints of mobile devices, which greatly limits the receiving sensitivity. Second, the commercial NFC transceivers are specially optimized for working in close distance with the target. We have designed an NFC sniffer for our experiments. Fig. 7.4 shows the block diagram of the sniffer, which consists of a 30 cm by 23 cm antenna, a pre-amplifier, and an ADC that is connected to a PC via USB to upload the collected samples. The NFC signal

overheard by the antenna is amplified and demodulated by the pre-amplifier and the AM demodulator, respectively. The signal is then digitalized by ADC and transmitted to PC for decoding. Our sniffer has a size of a tablet and average power consumption of 120 mW. Therefore, it can be easily connected to a mobile device via the micro USB interface to form a mobile sniffer. The NFC initiator devices used in this study include a Google Nexus 7 tablet, two smartphones (Google Galaxy Nexus and Samsung Galaxy Note 2), and an Adafruit PN532 NFC breakboard [2]. The NXP PN532 NFC chipset is adopted by the NFC breakboard, while all the other devices employ the NXP PN544 NFC chipset. These two chipsets are currently the most popular NFC chipsets used on commercial off-the-shelf mobile devices. Both chipsets use fixed transmission power which cannot be configured by software [20]. We use an NXP Mifare Classic tag as target.

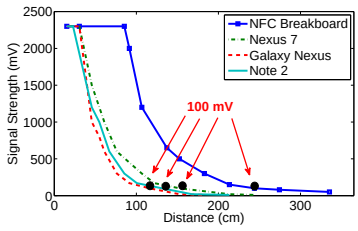


Figure 7.1: The received signal strength of the unattenuated signal over distance.

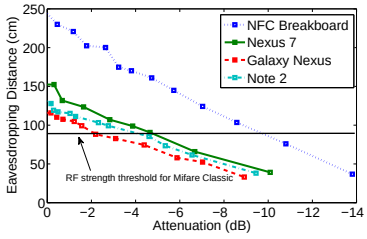


Figure 7.2: The received signal strength of the attenuated signal over distance.

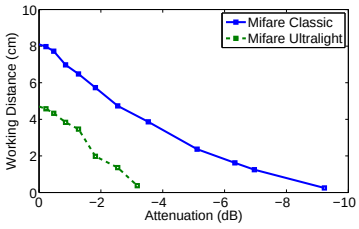


Figure 7.3: The maximum communication distances of two tags with different attenuation levels.

7.3.2 Results

In the first experiment, we measure the passive eavesdropping distances of both initiator and tag, without attenuating the RF field radiated by the initiator. We place the initiators on a desk, with the antennas of the devices facing forward. We activate one initiator at a time. The Mifare tag is placed in parallel and 1 cm from the antenna of the activated initiator. We place the sniffer near the initiator, and gradually move it away from the initiator.

Fig. 7.1 shows the signal strength of the initiators that is measured by the sniffer at different distances. As expected, the received signal strength decreases over distance. We can see that the signal is capped when the initiator-sniffer distance is short, as the output voltage of the sniffer can-

not exceed the voltage of its battery. We implemented a Miller decoder in Matlab to decode these samples. We find that the signal can be decoded if its strength is above 100 mV. When the strength is lower, the signal to noise ratio (SNR) is too low for successful decoding. As shown in Fig. 7.1, the 100 mV signal strength corresponds to physical distances of 152 cm, 131 cm, 116 cm, and 244 cm, respectively, when Nexus 7, Note 2, Galaxy Nexus, and Adafruit NFC breakboard are used as initiators. We are also able to decode the signal transmitted by the tag at maximum distances of 91 cm with Nexus 7, 85 cm with Note 2, 67 cm with Galaxy Nexus, and 121 cm with Adafruit NFC breakboard. Compared to the initiator transmissions, the eavesdropping distance of tag transmissions is significantly shorter, due to the much weaker signal strength of the tag response. We acknowledge that better hardware design and more advanced signal processing techniques could achieve even longer eavesdropping distances. Nevertheless, our results are already sufficient to demonstrate that the current NFC implementations on smartphone and tablet platforms are subject to passive eavesdropping from a distance at least an order of magnitude longer than the intended NFC communication range.

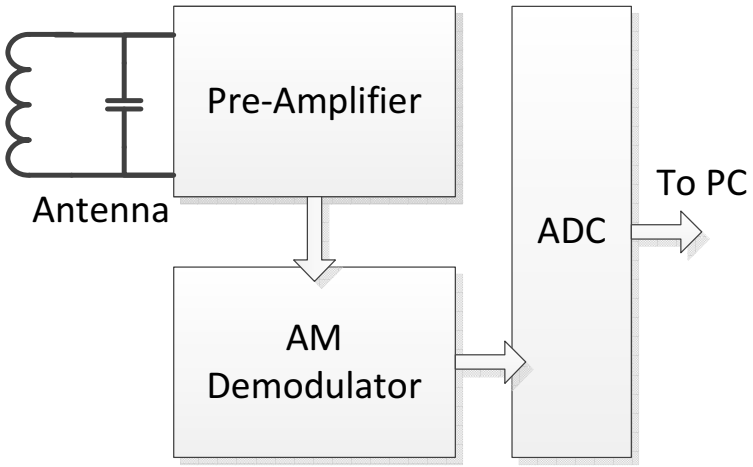


Figure 7.4: Block diagram of the NFC sniffer used in the measurement study.

A promising approach to defending against passive eavesdropping is to reduce the transmission power of the initiator. However, the current NFC chipsets adopt fixed transmission power, which leaves attenuating the signal externally the only choice. We need to answer the following two questions in order to design an external signal attenuator: 1) what is the maximum attenuation level that could be applied without sacrificing the reliability of data communication, and 2) what

is the resulted passive eavesdropping distance. We investigate these questions in the second experiment. We adopt the same experimental setting as in the first experiment, except that we cover the initiators with thin aluminum foils to attenuate the emitted RF field. The thickness and the area of the aluminum foil are adjusted to create different RF field strength, while the maximum passive eavesdropping distances are measured with our sniffer. We use a loop antenna connecting with an Agilent oscilloscope to measure the RF field strength after attenuation.

Fig. 7.2 shows that, as expected, for all the 4 tested initiators, the passive eavesdropping distances decrease when the attenuation level increases. When the strength of the NFC RF field is just enough to support reliable communication, our sniffer can achieve a maximum passive eavesdropping distance of around 80 cm, which is 67% (NFC Breakboard), 48% (Neuxs 7), 39% (Note 2), and 31% (Galaxy Nexus) shorter than those without attenuation. With such a short sniffing distance, the eavesdropping attack becomes significantly more difficult. However, the optimal attenuation level varies significantly for different initiators. Specifically, Fig. 7.2 shows that, to reduce the signal power to an undecodable level for sniffers, the NFC signal needs to be attenuated by 9.8 dB (NFC Breakboard), 5.9 dB (Neuxs 7), 4.2 dB (Note 2), and 2.2 dB (Galaxy Nexus), respectively. Such significant diversity is caused by the differences in initiator implementations, such as the size of antenna.

We now show that, for a given initiator, the maximum allowed attenuation level also varies significantly across targets. We measure the maximum communication distances between the NFC breakboard and two passive tags, Mifare Classic and Mifare Ultralight, with different attenuation levels applied to the RF field. Fig. 7.3 shows that the communication distances decrease when the attenuation level increases. However, the Mifare Classic can tolerate a maximum attenuation level of about 9 dB, while Mifare Ultralight can only tolerate about 3 dB. This huge difference is the result of the diverse receiving sensitivities of tags.

7.3.3 Discussion

We now summarize the results of our experimental study. First, current NFC implementations emit significantly more RF power than intended. As a result, the passive eavesdropping distance is at least an order of magnitude of the intended NFC communication range. This issue greatly increases NFC users' risk of being eavesdropped. Second, the NFC RF field strength can be effectively attenuated externally to enhance the security of NFC without sacrificing the communication reliability. However, the desired attenuation level varies significantly with the specific working conditions, including initiator transmission power, target reception sensitivity, initiator-target distance, and etc. Therefore, simple solutions such as an external signal attenuator with fixed amount of power reduction would not work for all scenarios.

These results have several important implications for the security of NFC systems. Properly implemented cryptosystems can offer strong security assurance even when the communication could be eavesdropped. However, as NFC is usually considered "physically secure", many upper-layer protocols of NFC applications do not implement encryption or only adopt short keys in encryption algorithms (such as DES [8]). With an passive eavesdropping distance up to 244 cm as shown in our study, these systems hence are exposed to malicious attacks. For instance, the leakage of pairing code during NFC-based Bluetooth paring could lead to possible passive eavesdropping or even man-in-the-middle attack on the following data communications. This issue is aggravated in active NFC communication scenarios, where both NFC devices actively transmits using high transmission power, and eavesdropping attacks on both of the devices could be launched over distance. Moreover, the feasibility of NFC eavesdropping attack renders encryption the last line of defense against attacks. Unfortunately, with the rapid advance of decryption techniques, many once considered "safe" encryption protocols, including WEP [30], DES [8], and RSA [24], have been demonstrated vulnerable when sufficient encrypted data is observed through eavesdropping.

7.4 Overview of nShield

7.4.1 Design Objectives and Challenges

It is shown in Section 7.3 that current NFC initiator implementations emit significantly more RF power than intended, which greatly increases the user's risk of being eavesdropped. This result motivates us to develop an NFC security protection device called nShield that dynamically regulates the strength of the RF field radiated by NFC initiators. nShield regulates the RF strength by absorbing the excessive RF power with its own antenna. nShield can be easily stuck on the back of mobile devices, and is solely powered by the absorbed RF energy, thus eliminating offline charging. Specifically, we have the following design objectives.

Adaptive RF field strength regulation. Today's NFC devices exhibit significant diversity in terms of initiator transmission power and the receiver sensitivity. nShield must be able to dynamically adjust the amount of absorbed power to ensure that the remaining RF power is just enough to sustain successful NFC communications. As nShield has no prior knowledge about the receiving sensitivity of the target, a "trial and error" approach is needed to determine whether NFC communications can be sustained at a particular power level. However, trying all possible attenuation levels incurs high delay due to the wide attenuation range and the low frequency of NFC transmissions.

Noninvasive operation. The operation of nShield should not rely on either initiator nor target. In other words, it should work in a standalone manner with no physical connections to neither initiator nor target. This requires nShield to be a self-sustained, self-powered device which has its own CPU and power source. Moreover, it should be transparent to the host, without the need to communicate with the host or modify the NFC protocols. The noninvasive and transparent nature of nShield enables it to easily retrofit the existing NFC devices with security protection. However, a key challenge presented by this design is that, as nShield cannot interact with either initiator or target, it has to determine the right transmission power solely based on the overheard transmissions.

Unintermittent protection. nShield should provide the host devices unintermittent protection against passive eavesdropping. In particular, the down time of protection caused by battery de-

pletion should be minimized. As discussed in Section 7.2, nShield scavenges energy from the NFC RF field, which is available only when the host device is active (e.g., when the screen of a smartphone is unlocked). When energy harvesting is not possible, nShield has to survive using the energy scavenged previously. Moreover, to keep the small form factor, nShield cannot adopt bulky high capacity batteries. Due to these challenges, nShield must minimize its power consumption as well as maximize the amount of power harvested from the host device. However, wireless charging is inherently inefficient [62], especially for peripherals like nShield that has tight cost budget and form factor constraints.

7.4.2 System Overview

nShield is composed of two major components, a software-defined passive NFC radio platform and an adaptive RF field attenuation algorithm. The software-defined platform is capable of receiving data from and transmitting data to NFC initiators, attenuating the NFC RF field using its antenna, and harvesting energy from the RF field. The adaptive attenuation algorithm dynamically determines the highest attenuation level that can still ensure communication reliability, according to the overheard NFC traffic. Fig. 7.5 shows the system architecture of nShield. An on-board MCU runs signal processing tasks such as encoding/decoding. nShield has two tuned loop antennas. The larger antenna is used for harvesting energy from the NFC initiator, as well as transmitting data to the initiator. The smaller antenna is responsible for overhearing data from the initiator. We show in Section 7.5 that, this dual antenna configuration is essential for maximizing the energy harvesting efficiency without sacrificing the receiving performance, as the receiving antenna and the harvesting antenna require fundamentally different design methods.

The harvesting antenna is connected with an RF bridge rectifier, which rectifies the RF signal to a DC voltage. The DC voltage is then regulated to provide power to the system and charge a 20 mAh on-board battery. In Section 7.5 we show that the voltage matching between the harvesting antenna and the battery plays a critical role in maximizing the amount of power harvested by the system. The load modulator is connected with the rectifier, which alters the load of the harvesting

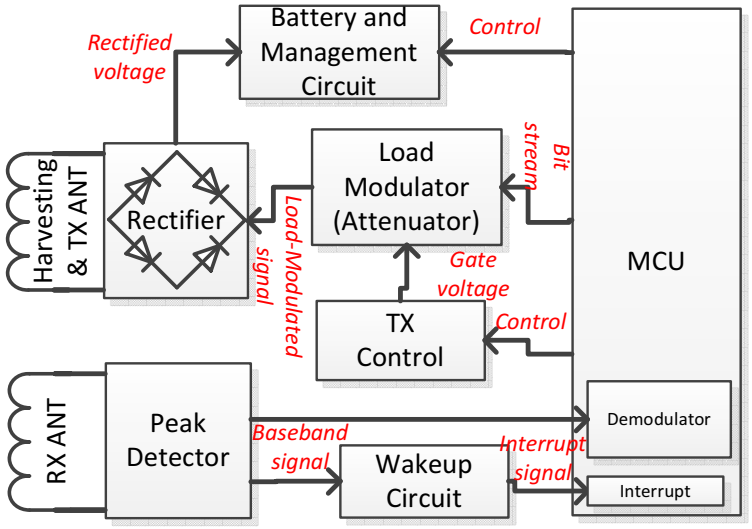


Figure 7.5: Block Diagram of nShield.

antenna to transmit data to the NFC initiator. Since the load modulation-based communication scheme adopted by NFC standard requires strict timing, nShield employs a hardware TX control circuit to accurately generate the clock used by the load modulation and precisely synchronize the data to be transmitted. The TX control circuit can generate different clock frequencies according to the data rates of the modulation schemes. nShield reduces the risk of eavesdropping by absorbing the excessive RF power radiated by the initiator with an adjustable attenuator, which is multiplexed with the load modulator.

The receiving antenna is connected to a peak detector, which removes the AM carrier from the RF signal. The hardware-based demodulator on the MCU demodulates the baseband signal, from which the raw data is retrieved. A key novelty in the design of nShield is to exploit the hand-shake mechanism in the target discovery process to determine the optimal transmission power of the initiator. Specifically, nShield infers whether the previous messages are successfully received by examining the logical relationship between consecutive initiator messages. To reduce the delay of determining the optimal attenuation level, nShield adopts a binary search algorithm to accelerate the search. nShield falls asleep to conserve energy when no NFC signal is present. A low-power wakeup circuit connected with the peak detector generates an interrupt signal to wake up the system once NFC RF field is present.

Fig. 7.6 shows a prototype system of nShield. The size of the circuit board and the antenna is 5.5 cm by 5.3 cm and 9.6 cm by 9.6 cm, respectively. We note that this antenna is specially designed for Nexus 7 tablet. The size of antenna can be reduced for smartphones, without sacrificing the energy harvesting efficiency and attenuation performance. The size of the prototype circuit board can be shrunk significantly by removing unnecessary components like debug port, buttons and LEDs. As a result, nShield can be easily fit on diminutive thin-film circuit boards, which could be stuck to the back of small-size mobile devices. The total component cost of our prototype implementation is under $20, and could be further reduced when nShield is mass-manufactured.

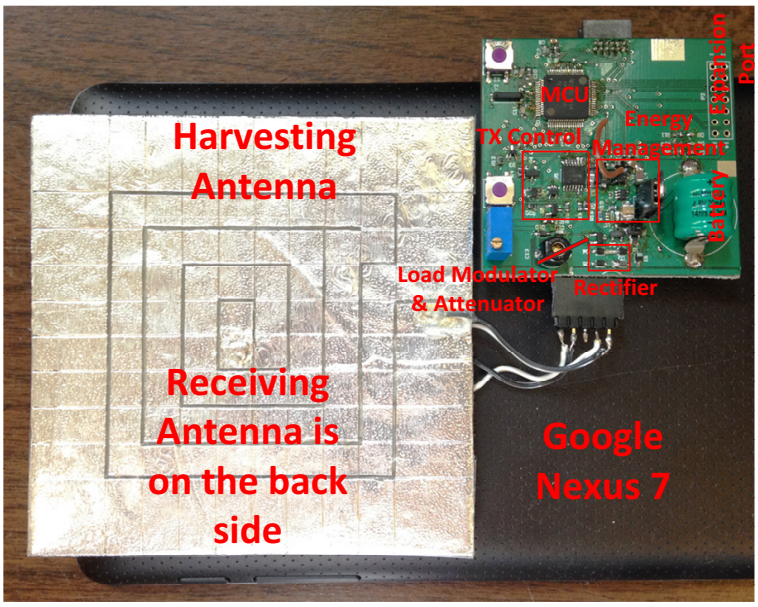


Figure 7.6: Antenna and circuit of nShield mounted on the back of a Google Nexus 7 tablet.

7.5 Maximizing Harvested Energy

nShield is powered solely by the energy harvested from NFC transmissions. The capability of harvesting a large amount of power not only enables the uninterrupted protection of nShield, but also helps increase the attenuation range of the host's NFC transmission power. Fig. 7.7 shows the block diagram of the energy harvesting subsystem of nShield, which comprises a harvesting antenna and an energy management circuit. These two components work together to determine the amount of power that could be harvested. We show that they must be carefully designed to

maximize the harvested power. We define the following two terms to characterize the performance of energy harvesting. *Energy (power) transfer efficiency* is defined as the ratio of the amount of energy (power) transferred to the harvesting antenna, to the amount of energy (power) transmitted by the NFC initiator. *Energy (power) harvesting efficiency* is defined as the ratio of the amount of energy (power) transferred to the receiving system after rectifying and regulation, to the amount of energy (power) transmitted by the NFC initiator. Obviously, for any wireless power transfer system, energy (power) harvesting efficiency is always lower than energy (power) transfer efficiency.

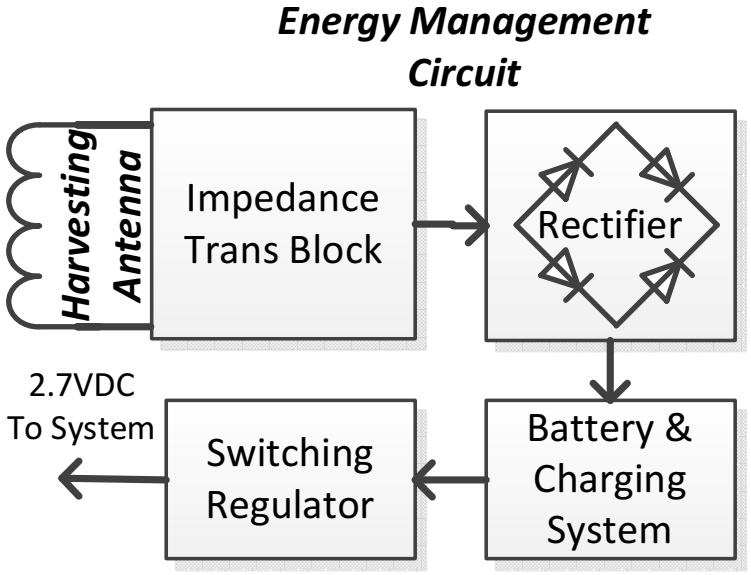


Figure 7.7: Block Diagram of energy management circuit on nShield.

7.5.1 Harvesting Antenna

When the communication between an NFC initiator and a target device commences, energy transfers from the transmitting antenna to the harvesting antenna via resonant inductive coupling [58] through air. The NFC antennas are essentially inductors, which have inductance as well as series resistance. The radiation efficiency of NFC antennas can be quantified using *quality factor* (or Q-factor), which is the ratio of the inductive reactance to the series resistance of the antenna at 13.56 MHz:

$$Q = \frac{\omega L}{R} = \frac{27.12\pi L}{R} 10^6 \tag{7.1}$$

where ω is the working frequency of the antenna, and L and R are the inductance and the series resistance of the antenna, respectively. The Q-factors of the transmitter antenna and the harvesting antenna largely determine the power harvesting efficiency between antennas. Given the Q-factors of transmitter antenna, Q_t, and the harvesting antenna, Q_h, the maximum power transfer efficiency of the NFC antenna pairs can be expressed as [58]:

$$\Pi_{max} = \frac{U^2}{(1+\sqrt{1+U^2})^2} \tag{7.2}$$

$$U = k\sqrt{Q_t Q_h} \tag{7.3}$$

where k is the coupling coefficient, with 0 being completely uncoupled and 1 being perfectly coupled. k depends on many factors such as the distance between the two antennas, antenna alignment, and etc. For NFC, since the communication pairs are always placed in proximity, k is usually above 0.1 [17]. For each nShield installation, k is largely a constant value, as nShield is fixed on the back of the mobile device. Due to the NFC communication bandwidth requirement (about 1.8 MHz [17]), the Q-factor of the transmitting antennas, Q_t, is about 15 for most NFC devices [29]. As a result, the maximum power transfer efficiency of nShield is largely determined by the Q-factor of the harvesting antenna, Q_h. A high power transfer efficiency can thus be achieved by using harvesting antennas with high Q-factors (above 50). For example, if k, Q_t, and Q_h of an NFC energy harvesting system are 0.2, 15, and 100 respectively, a maximum power transfer efficiency of 77% could be achieved. A key insight of this analysis is that, the harvesting antenna cannot be reused by the NFC transceiver, due to the conflicting requirements of the Q-factors. Therefore, to support efficient energy harvesting and reliable NFC communication at the same time, a dual antenna configuration (one high Q-factor antenna and one low Q-factor antenna) must be adopted.

According to (7.1), to improve Q-factor of an NFC antenna, we can either increase its inductance or decrease its series resistance. In our harvesting antenna design shown in Fig. 7.6, we use wide antenna tracks to decrease the series resistance, and closely couple the antenna tracks to increase the inductance. The parasitic capacitance also contributes to the series resistance of the

antenna. We adopt a single layer antenna to decease the parasitic capacitance. The resulted high Q-factor ensures that, when the transmitter antenna and the harvesting antenna are closely coupled, the harvesting antenna can receive most of the radiated energy. The implementation details of the harvesting antenna are given in Section 7.7.

7.5.2 Energy Management Circuit

Another major factor that affects the amount of power harvested to the system is the design of the energy management circuit. The energy received by the harvesting antenna has to be transferred to the energy storage components in the system, e.g., batteries or super capacitors. A common practice for maximizing power transfer is to match the output impedance of the antenna with the input impedance of the load [57]. The maximum power that can be transferred, P_{load}, can be expressed as:

$$P_{load} = \left(\frac{U_{ant-open}}{R_{ant}+R_{load}}\right)^2 R_{load} = \frac{U_{ant}^2}{4R_{ant}} = 0.25P_{max} \tag{7.4}$$

where $U_{ant-open}$ is the open-circuit root-mean-square voltage inducted on the harvesting antenna, R_{ant} and R_{load} are the impedances of the antenna and the load, respectively, and P_{max} is the maximum power that the harvesting antenna can receive. We can see that P_{load} equals a quarter of P_{max}, when and only when $R_{load} = R_{ant}$.

However, the perfect impedance matching is impossible for energy harvesting systems, since the input impedance of the energy management circuit, R_{load}, varies significantly with the system load. To solve this problem, instead of matching impedance, nShield employs *voltage matching*. Since R_{ant} and R_{load} are in series, when $R_{ant} = R_{load}$, the voltage across R_{ant} and R_{load}, denoted as U_{ant} and U_{load}, respectively, are also identical, i.e., $U_{ant} = U_{load} = 0.5U_{ant-open}$. Therefore, an alternative way to achieve the maximum power transfer is to match U_{load} to $0.5U_{ant-open}$. Since $U_{ant-open}$ is a constant value when the harvesting antenna is attached to the initiator, the maximum power transfer can be achieved by letting $U_{load} = 0.5U_{ant-open}$. A key question is how to stabilize U_{load} when system load varies. nShield connects the battery directly to the output of the rectifier,

which makes U_{load} stay equal to the voltage of the battery, U_{bat}. Since most batteries have stable output voltage regardless the discharging level and the output current (system load), the optimal energy transfer rate can be always maintained.

However, $0.5U_{ant-open}$ could be difficult to match with U_{bat} in practice, as the harvesting antenna and the energy management circuit are usually separately designed to meet different requirements (e.g., Q-factor, system power consumption, system voltage, etc.). An impedance transformation block, such as L-section circuit or RF transformer [12], can be employed to shift $U_{ant-open}$ to a given voltage. Although an impedance transformation block is not required by our current implementation of nShield, it would be required if nShield employs a Lithium battery (3.6 V). It is also worth noting that, super capacitors are ill-suited for nShield, as their output voltages vary significantly with the discharging levels. To protect the batteries, we use a linear regulator and MOSFET switches to manage the charging. We do not use a switching regulator since it tends to alter the voltage matching point thus reduces the energy harvesting efficiency. Fig. 7.7 shows the design of energy management circuit of nShield. Our experiment in Section 7.8.1 shows that nShield can harvest 55 mW power constantly from the NFC initiators on typical smartphones.

7.5.3 Tag Emulation

As discussed in Section 7.2, the initiator adopts a low probing rate [50] when no target device is nearby, which only allows limited amount of energy to be harvested. Nevertheless, we show in Section 7.8.2 that, as long as the host device is active for more than 429 seconds/day, the energy harvested during the probing phase is sufficient for keeping the battery charged. In the rare case when the mobile device is only infrequently unlocked for a long period, nShield may deplete its battery. To address this issue, we adopt a technique called tag emulation to have the initiator significantly increase its duty-cycle. Specifically, nShield emulates itself as a passive ISO14443A tag and responds to the probing messages sent by the initiator. As a result, it triggers the initiator to stay active. This leads to a 10X increase of the initiator output energy, allowing nShield to be rapidly charged. However, this process may interfere with NFC transactions, as the initiator cannot

communicate with other target devices when the tag emulation is active. We adopt the following adaptive mechanism to address this issue. First, nShield pauses the tag emulation for 1 second every 2 seconds, allowing the initiator to discover other target devices during the pause. Second, nShield only activates tag emulation when the discharging level of the onboard battery is lower than 30%.

7.6 Adaptive RF Field Attenuation

7.6.1 Attenuator

nShield reduces the risk of being eavesdropped by attenuating the NFC RF field strength using the harvesting antenna. The level of attenuation to the RF field is adjusted by the load of the harvesting antenna. nShield adopts a MOSFET as the variable load, i.e., attenuator to the antenna. The resistance of the MOSFET is controlled by its gate terminal voltage, which is dynamically set by the adaptive RF field attenuation algorithm described in Section 7.6.2, using an onboard DAC. A novel design of nShield is that the attenuator is multiplexed with the load modulator of the NFC transmitter. This design reduces the cost and size of nShield. Our experiment in Section 7.8.4 shows that nShield can achieve an attenuation range of 10.86 dB, which is sufficient for the purpose of regulating NFC RF field strength.

7.6.2 Adaptive RF Field Attenuation Algorithm

nShield adapts the signal attenuation level dynamically to ensure reliable communication between the initiator and the target device. nShield equally divides the whole attenuation range into N discrete levels. The goal of adaptive RF field attenuation is to find the optimal attenuation level in the N levels, with which the attenuated field strength is just enough to support reliable bi-directional communications between the initiator and the target. Fig. 7.8 illustrates the relationship between Packet Reception Ratio (PRR) and the attenuation levels (AL). nShield tries to use an attenuation

level as high as possible, while ensuring the resulted PRR to be close to 1, i.e., high communication reliability. A_{opt} shown on Fig. 7.8 is the optimal attenuation level.

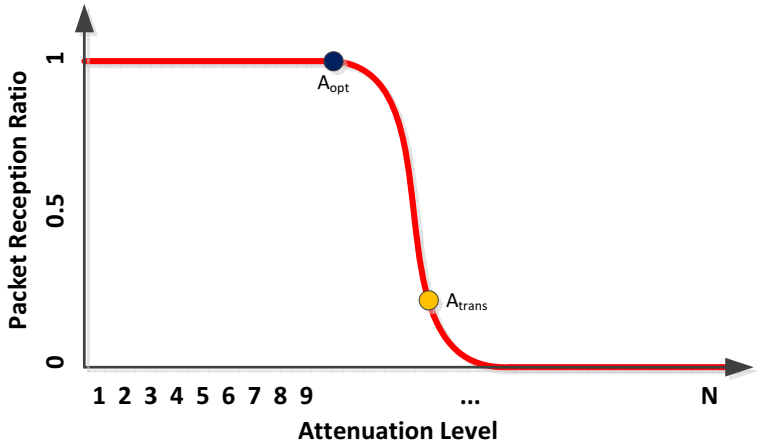


Figure 7.8: An illustration of the attenuation level vs Packet Reception Ratio relationship.

However, a key challenge in the design of nShield is that, without prior knowledge about the target device, such as reception sensitivity and initiator-target distance, nShield cannot know what RF field strength would support reliable communications. NFC work in a poll-response fashion, in which the target only transmits after it was polled by a message from initiator. We refer to the process of a polling and its subsequent response as a *polling round*. To find out wether an attenuated field strength can support bi-directional communication, the initiator has to attempt a polling round with the attenuation level in question. nShield learns if a polling round is successfully complete, by examining the logic of the polling messages of consecutive polling rounds. In particular, some polling messages, such as the Single Device Detection Request and the Select Request defined in the NFC-A standard, can only be transmitted if the previous polling round succeeds. When overhearing such polling messages, nShield infers that the previous polling round ends successfully.

As shown in Section 7.8.3, for the passive communication mode, the field strength required for completing the first polling round is lower than that for completing later polling rounds. This phenomenon is caused by insufficient energy left on the tag after the first polling round. Passive tags rely on the energy from the NFC RF field to operate. After activating the RF field, the initiator pauses for certain time to charge the tag before starting the first polling round. The length of this charging period is usually much longer than the interval between consecutive polling rounds.

Even if the RF field strength was not sufficient to sustain the successive polling, the first polling round may still succeed due to the energy harvested from the initial charging period. As a result, for passive communication mode, the success of the first polling round after the activation of the RF field is not a good indicator if the field strength is strong enough for sustaining bi-directional communication. In our design, we deem a field strength sufficient only if it can support the first three consecutive polling rounds.

Algorithm 3 *Adaptive RF Field Attenuation*

Input: N: number of attenuation levels.
Output: n_{opt}: optimal attenuation level.
Used sub-function: $Comm(n_i)$: attempt communication with attenuation level n_i. This sub-function returns "success" only if the first three polling rounds are completed successfully with the attenuation level n_i

$N_{upper} = N$
$N_{lower} = 1$
$n_{opt} = N/2$
while $N_{upper} - N_{lower} > 2$ **do**
  if $Comm(n_{opt}) = success$ **then**
    $N_{upper} = round((N_{upper} + n_{opt})/2)$
  else
    $N_{lower} = n_{opt}$
  end if
  $n_{opt} = round((N_{upper} + N_{lower})/2)$
end while
return n_{opt}

An interesting question is that, with N different attenuation levels, in what order should nShield attempt communications. A naive solution is to attempt with all N levels from a high-to-low or low-to-high order, until an attenuation level for supporting reliable bidirectional communication is found. However, this approach incurs high delay (at least several seconds). We adopt the Binary Search Algorithm (BSA) to accelerate the search process. With BSA, the search starts from the middle of all attenuation levels. Depending on whether the following polling rounds are successful or not, BSA discards the lower or higher half of the levels that unlikely contain the optimal level. For example, if any of the three following polling round fails, BSA discards all the levels that are higher than the currently attempted level. BSA repeats this process with the remaining levels

until there is only one level left. However, due to the transition region on the PRR-AL curves (see Section 7.8.3), BSA may fail to locate the optimal attenuation level. This is because whether an attenuation level in the transition region, such as A_{trans} on Fig. 7.8, can support a successful polling round is probabilistic. When the polling rounds attempted with A_{trans} succeed, all the attenuation levels higher than A_{trans}, including the optimal level A_{opt}, would be discarded. To address this issue, we adopt a modified BSA in nShield. It works in the same way as the original BSA, except that it only discards half of the higher levels after three successful polling rounds. As the transition region of the PRR-AL curve is very narrow (see Section 7.8.3), this ensures that the optimal level would not be accidentally discarded. Algorithm. 3 shows the pseudo-code of the adaptive RF field attenuation algorithm.

nShield exploits the target discovery process, which is always performed by the initiator in the initial phase of the communication, to perform adaptive RF field attenuation. The NFC initiator periodically performs this process by broadcasting NFC discovering probes (at a rate about 3Hz on Android smartphones). If a target NFC device (which can be a tag or another NFC initiator working in active mode) hears this probe, it will send an acknowledgement message back to the initiator. The initiator will then confirm the discovery of the target device by broadcasting a response. The two devices will then exchange a few messages back and forth to learn a few parameters (such as IDs and capabilities). There are several advantages of exploiting this process for adaptive RF field attenuation. First, the NFC target discovery process is mandatory in all NFC communication modes and NFC standards (NFC-A, NFC-B, and NFC-F) [18]. Second, this process does not involve the data payload. The communication conducted during adaptive RF field attenuation might be eavesdropped, due to the possibly high initiator transmission power. However this does not lead to security breach since there is no data payload exchange. Once adaptive RF field attenuation is done, the following data communication is protected from passive eavesdropping. If the adaptive RF field attenuation is not finished yet in the last phase of the target discovery process, nShield will jam the communication to force the initiator to restart the process.

7.7 Implementation

We implemented a prototype nShield, which is shown in Fig. 7.6. We use a TI MSP430F2618 as the MCU on nShield. It integrates many low-power components used by nShield, such as comparator, ADC, DAC, and DMA controller. A 4.8 V 20 mAH NiMh battery is adopted to store the harvested energy.

We implement the harvesting antenna using layered tapes and aluminum foil. To maximize the attenuation range, the size of the harvesting antenna should be slightly larger than the antenna on the NFC initiator, so that all magnetic flux generated by the initiator would undergo the attenuation before reaching the target. For example, our prototype antenna attached to Nexus 7 has a dimension of 9.6 cm by 9.6 cm, slightly larger than the NFC antenna in Nexus 7. We build the base of the antenna using 2mm thick layered tapes. We apply the aluminum foil to one side of the base, and cut the foils into 7 mm wide tracks to reduce the series resistance. The tightly coupled tracks increase the inductance of the antenna. The combination of high inductance and low series resistance leads to a high Q-factor (> 100), which is essential for achieving high energy transfer efficiency. The NFC signal reception antenna is prototyped using the same materials and techniques, except that it has much thinner tracks. We use an impedance analyzer to tune the Q-factor of the antenna to the optimal value of 15 [29]. The harvesting and receiving antennas are then glued together. The two prototype antennas can be easily mass-manufactured using flexible thin film circuits.

We implement an NFC transceiver on nShield. The reception path is composed of a peak detector, a comparator, and a software decoder. The RF signal from the antenna is first converted to baseband signal by the peak detector, and then converted to clean logic levels by the comparator. The decoder is implemented in software on the MCU. To decrease the computational overhead, hardware components on the MCU are adopted to assist the decoding. Specifically, a hardware timer is adopted to timestamp the transitions of the logic levels, and a DMA controller is employed to automatically transfer the timestamps to the RAM. This design automatically collects samples without software intervention, enabling low power asynchronous decoding. The data is then verified using CRC and reported to upper layer protocols. For transmission, nShield adopts

the load modulation communication techniques [18], in which the load of the antenna is modulated according to the data to be transmitted. We adopt a high speed MOSFET (Fairchild FDV301N) as the load modulator (multiplexed with attenuator), which can be easily driven by the onboard DAC due to its very low gate driving voltage (less than 1 V). The bridge rectifier is implemented by four NXP PMEG600 low forward drop Schottky diodes to minimize the energy loss on rectifying. To generate accurate baud rates and subcarrier frequencies, a 13.56 MHz crystal oscillator and a hardware clock divider are employed. We implemented the ISO14443A (NFC-A) protocol on nShield, which supports a data rate of 106 kbps. Since the modulation/demodulation tasks are mainly handled by hardware, higher data rates can also be easily supported by nShield. Moreover, since many protocols are implemented in software, nShield can be easily customized to meet the requirements of different applications. As a software-defined radio platform, nShield can also be configured to provide malicious content protection functions [50].

nShield employs several techniques to optimize its power consumption. For example, at runtime, unused components are shut down. The clock rate of the MCU is also dynamically adjusted according to the workload. To further reduce power consumption, nShield enters sleep state when no NFC RF field is detected. During sleep, all onboard components except the low-power time keeping timer are shut down.

7.8 Experimentation

this section, we study the performance of nShield using a set of experiments. We adopt two initiators (Google Nexus 7 tablet and Adafruit PN532 breakboard) and two tags (Mifare Classic and Mifare Ultralight). We choose these devices not only because they are representative NFC devices on the market, but also due to their diverse characteristics. For example, the Adafruit PN532 breakboard has a large antenna and can transmit a large amount of power (about 450 mW), while Google Nexus 7 has a much smaller antenna and much lower transmission power (about 200 mW). The Mifare Classic tag has an antenna size of a credit card which is very common among passive tags, while Mifare Ultralight only has an antenna size of a coin, which is considered to be a “weak” tag.

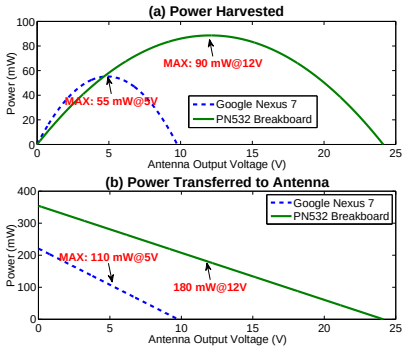


Figure 7.9: Power transferred and harvested from Nexus 7 and PN532 breakboard.

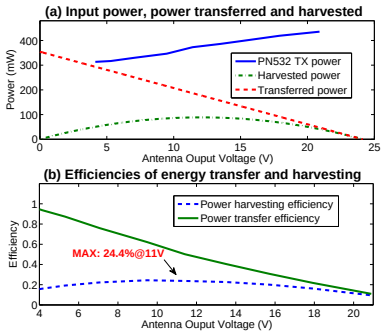


Figure 7.10: Power harvesting efficiency and power transfer efficiency.

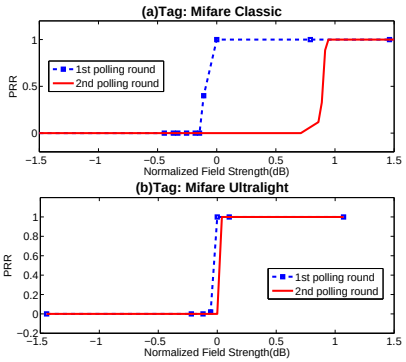


Figure 7.11: PRR-FS curves of two NFC tags

The testing equipments we use include an Agilent DSOX2024 oscilloscope, an Agilent 34410A benchtop multimeter, an Extech handheld multimeter, and an SDR-Kits VNWA3 Vector Network Analyzer.

7.8.1 Amount of Harvested Power

We measure the amount of power that can be harvested by nShield, and the power transfer and harvesting efficiency with two experiments in this subsection.

In the first experiment, we employ both of the initiators for testing. The harvesting antenna (shown in Fig. 7.6) is attached to the back of Google Nexus 7, and to the surface of the PCB antenna on PN532 breakboard. We connect a potentiometer to the antenna as the load. The output voltage and current of the antenna under different loads are measured with an Agilent 34410A benchtop multimeter. A linear regression is applied to the results to compute the internal resistances and the open-circuit output voltages of the harvesting antenna. We then compute the power harvested by the system and the power transferred to harvesting antenna under different loads.

Fig. 7.9 (a) depicts the harvested power under different antenna output voltages. We can see that the curves are parabolas, with the maximum power of 55 mW at 5 V, and 90 mW at 12 V, respectively, when Google Nexus 7 and PN532 breakboard are used. The amount of power that can be harvested from PN532 breakboard nearly doubles that from Google Nexus 7. This is because PN532 breakboard has a much higher transmission power than Nexus 7, according to our

measurement. However, as the antenna is optimized for working with Nexus 7, nShield cannot harvest the maximum amount of power from PN532 breakboard. In particular, the maximum power is harvested at 12 V output and the battery voltage on nShield is only 4.8 V. This voltage mismatch limits the maximum harvested power to be only 57 mW. An impedance matching block is required to shift the open-circuit voltage to around 10 V for PN532 breakboard, as discussed in Section 7.5.2. On the other hand, nShield can receive the maximum power when working with Nexus 7, due to the tight voltage matching. These results also confirm that a super capacitor is a poor choice for energy storage on nShield, since the voltage of super capacitors varies significantly with its discharging level, resulting a poor voltage matching.

Fig. 7.9 (b) shows the power transferred to the harvesting antenna at different output voltages. We can see that the transferred power decreases linearly when the output voltage increases. When the output voltage of the harvesting antenna is zero, the antenna receives the maximum power. However, it also delivers virtually no power to the system, resulting in an extremely low power harvesting efficiency, as observed from both Fig. 7.9 (a) and (b). When the output voltage is about half of the antenna open-circuit voltage, the maximum power is harvested, although the power transferred to the antenna is significantly lower. These results show that, in order to deliver the maximum power to the system, the battery and the harvesting antenna must achieve a voltage matching.

We next evaluate the power harvesting and transfer efficiencies of nShield. We only use Adafruit PN532 breakboard as initiator in this experiment because the transmission power of Nexus 7 cannot be accurately measured due to its packaging. The transmission power of the PN532 board can be obtained by measuring the current draw on the TVDD pin of PN532 chip, which supplies power to its internal coil exciting circuits. The harvesting antenna is connected with a potentiometer which serves as a variable load.

Fig. 7.10 (a) shows the amount of power transmitted, transferred, and harvested, under different loads to the harvesting antenna. We can see that the transmission power increases when the load becomes lighter. The change of the transmission power is due to the detuning effect, in which the

tuning of the initiator's antenna is varied by the mutual coupling between the harvesting antenna and the initiator antenna. A heavier (lighter) load to the harvesting antenna creates a slightly stronger (weaker) mutual coupling, which in turn leads to a stronger (weaker) detuning effect. The detuning effect changes the impedance of the antenna, resulting in less power transferred. The highest transmission power is about 440 mW.

Fig. 7.10 (b) shows the computed energy transfer and harvesting efficiencies. We can observe that the energy transfer efficiency increases linearly with the load to the harvesting antenna, while the energy harvesting efficiency is a parabola curve which peaks at the voltage matching point (11 V). When the output voltage of the harvesting antenna is below 4 V, the energy transfer efficiency is close to 1. At this point, most of the transmitted energy is absorbed by the harvesting antenna, and the strength of the RF field created by the initiator is significantly attenuated. The energy harvesting efficiency peaks at 24.4% when the output voltage of the harvesting antenna is 11 V. We discuss the energy harvesting efficiency in Section 7.9.

7.8.2 System Power Consumption and Lifetime

We use an Agilent 34410A benchtop multimeter to measure the power consumption of nShield. The results are summarized in Tab. 7.1. The most power consuming states are data reception and transmission. This is because the MCU has to work at a higher system clock rate to meet the strict timing requirements of the NFC data reception and transmission, and several system components (e.g., TX control circuit) need to be powered on. Although the idle/RX/TX power consumption are high, their impact on system lifetime is actually insignificant, since nShield spends most of the time in the sleep state with a power consumption of only 23 uW. This is due to the fact that, the NFC initiator is usually inactive most of the time (e.g., when the mobile device is locked), during which nShield is asleep.

Thanks to the large amount of power harvested from NFC transmissions and low power design, nShield can sustain its operation solely on the harvested energy. NFC standard requires initiators to insert long guard time between consecutive polling rounds [18]. As a result, NFC initiators are

Sleep	Idle listening	RX	TX	Attenuation
23 uW	8.7 mW	13.1 mW	18.1 mw	9.8 mW

Table 7.1: System power consumption under different states.

in idle listening most of the time when activated. This causes nShield to be idle during most of its active period, leading to an average active power consumption of 8.7 mW. As nShield can harvest 55 mW power from an active NFC initiator, it maintains a net power gain of 46.3 mW during its active state. For typical Android devices, the integrated NFC initiators are duty-cycled at 10% [50] during probing. With its low sleep power consumption, the battery on nShield can stay fully charged if the mobile device is unlocked for average 429 seconds per day, which can be met by smartphones and tablets in most circumstances [46][4]. When the discharging level of the onboard battery is low, nShield automatically activates tag emulation, which increases the charging rate by 10X to rapidly charge the battery. Moreover, even when energy harvesting is not possible (e.g., NFC is disabled), the lifetime of a fully charged nShield still exceeds one month, thanks to its low sleep power consumption.

The above results show that nShield's capability of harvesting high amount of power plays a significant role in achieving the perpetual operation. As nShield can be only charged when the screen of the device is unlocked, the minimum harvested power for sustaining nShield depends on how the users interact with mobile devices. A recent survey [65] shows that on average U.S. users spend 58 minutes on smartphones per day, which is more than enough for nShield to stay fully charged. However, for light smartphone users, the harvesting power should be sufficiently high. Compared to EnGarde whose harvested power is only about 30 mW[1], nShield decreases the minimum active time of the phone by more than 50% (7.15 min vs 15.5 min).

[1] The exact amount of harvested power is not given in [50]. However, it is expected to be much lower than 30 mW, due to the load-source mismatch and the loss on rectifying and regulating components.

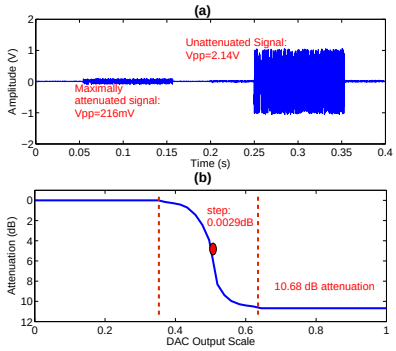


Figure 7.12: nShield achieves an attenuation range of about 10dB.

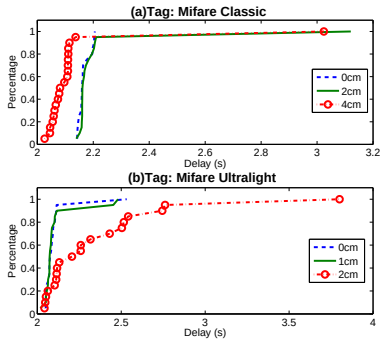


Figure 7.13: Delay caused by determining attenuation level.

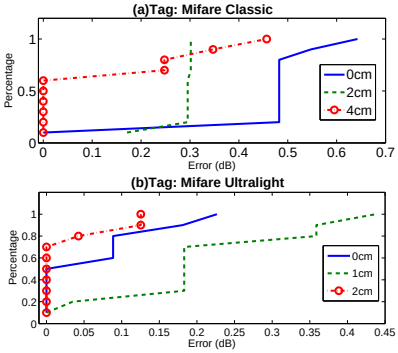


Figure 7.14: Accuracy of attenuation level determined by nShield.

7.8.3 Receiver Characteristics

In this subsection, we study the receiving characteristics of passive NFC tags, by measuring the PRR-FS (Packet Reception Ratio vs Field Strength) curves. The purpose of this experiment is to show two key observations based on which the adaptive RF field attenuation algorithm is designed: 1), the transition regions on the PRR-FS curves are very narrow, and 2), the field strength required for completing the first polling round is higher than the subsequent rounds.

We attach a thin aluminum antenna to the back of each tag to measure the field strength, using an Agilent DSOX2024A oscilloscope. A Nexus 7 serves as the NFC initiator in this experiment. We vary the field strength near the tag by changing the distance between the initiator and the tag. The PRR associated with each field strength value is computed from 100 transmissions. The field strength measurements are normalized.

Fig. 7.11 (a) and (b) show the PRR-FS curves of Mifare Classic tag and Mifare Ultrlight tag, respectively. We can see that, all the curves have narrow transition regions (<0.2 dB) in which the PRR values quickly increase from 0 to 1. We further observe that, Mifare Ultralight tag has a narrower transition region than the Mifare Classic tag (0.05 dB vs 0.2 dB). This is because the Mifare Ultralight tag has a much smaller antenna size, making it more sensitive to the field strength. For each tag, we can see that the field strength required for a successful first polling round is lower than that for the second polling round. As mentioned in Section 7.6.2, this is due to the fact that

the tag has more time to harvest energy before the first round of polling.

7.8.4 Attenuation Range and Granularity

nShield provides a wide attenuation range and fine attenuation granularity, which allows it to precisely control the strength of the NFC RF field to the optimal level. This subsection evaluates the attenuation range and step that can be achieved by nShield. We manually tune the DAC connected with the attenuator to sweep through its entire voltage output range with a step of 0.05 V. To measure the attenuated signal strength, we use an Agilent probe to form a small loop antenna, and connect the probe to an Agilent DSOX2024A oscilloscope. We record the measured peak-to-peak amplitude (Vpp) of the NFC signal.

Fig. 7.12 (a) depicts the signals that are maximally attenuated and unattenuated. We can see that nShield can significantly decrease the strength of NFC signals, as the Vpp of the signal decreases from 2.14 V to only 0.216 V after the maximum attenuation level is applied. Fig. 7.12 (b) shows the computed attenuation levels with different DAC output. We can observe that the effective attenuation region roughly takes about a quarter of the full output scale of the DAC, ranging from 0.8 V to 1.4 V. This is due to the characteristic of the attenuator on nShield, which is a high-speed switching MOSFET. The MOSFET is completely shut down when the gate voltage is below 0.8 V, and is saturated when the gate voltage is above 1.4 V. Therefore, it operates as a variable attenuator only when the gate voltage is between 0.8 V and 1.4 V. The maximum attenuation, 10.86 dB, is achieved when the MOSFET is saturated. We can also observe that the attenuation is nonlinear with the DAC output, resulting in a nonconstant attenuation steps. The maximum step occurs when the MOSFET operates near the middle of the effective attenuation region. For a 16 bit DAC with 2.3 V reference, the maximum step is 0.0029 dB. The wide attenuation range and fine attenuation step allows nShield to precisely attenuate the RF field with wide strength range to the optimal level. This ensures nShield to best protect the security of NFC while maintaining reliable communication.

7.8.5 Delay of Adaptive Attenuation

The delay caused by the adaptive attenuation algorithm is a critical performance metric for nShield, since a long delay would have significant impact on the user's experience. In this section, we measure the delay introduced by the adaptive attenuation algorithm, using a Mifare Classic tag and a Mifare Ultralight tag. We define the delay as the interval from the time instant when the initiator sends the first probe to the tag to the time instant when the optimal attenuation level is determined. We use the hardware timer on nShield to timestamp these events and measure the delay. For each tag, we measure the delay associated with 3 different optimal attenuation levels, by varying the tag-initiator distances. To illustrate the delay in practical settings, we hold the tags with hands, which introduces small tag-initiator distance variations during communications. We repeat the experiment at each distance for 20 times.

Fig. 7.13 shows that, most of the delays fall below 2.2 s, while the mean delay is 2.1 s. An interesting phenomenon is that the delay of Mifare Classic incurred at a distance of 4 cm is smaller than those incurred at 2 cm and 0 cm. This is because, the delay is largely proportional to the number of steps that the adaptive attenuation algorithm has to take to find the optimal attenuation level, which varies between 6 and 12 in nShield. Thus a longer communication distance could possibly incur a shorter delay. We also notice that the adaptive attenuation algorithm is resilient to minor tag-initiator distance variation, as nShield can almost always find the optimal attenuation level within 2.2 seconds. We did observe some long delays (3s to 4s), although they are rare ($< 5\%$). Our further investigation indicates that they are caused by occasional initiator halts, in which the initiator pauses its transmission for 1 to 2 seconds, while the RF field remaining active. Finding the exact reason of this long initiator halt is left for future work.

7.8.6 Accuracy and Effectiveness of Adaptive Attenuation

We evaluate the accuracy of adaptive attenuation algorithm in estimating the optimal attenuation level in this subsection. The initiator we use in this experiment is the PN532 breakboard. For each tag under test, we evaluate the optimal attenuation level with different tag-initiator distances. We

define the optimal attenuation level as the highest attenuation setting that can support successful initiator-tag communications for 10 seconds. We manually determine the ground-truth optimal attenuation level for each tag-initiator distance, by examining all attenuation levels from a high to low order. We use an Agilent probe to form a small loop antenna, and connect the probe to an Agilent DSOX2024A oscilloscope to measure the attenuated RF field strength. We then run the adaptive attenuation algorithm for ten times, and measure the resulted RF field strength of each run.

Fig. 7.14 shows that, 90% of the estimation errors of the Mifare Classic tag at distances of 0 cm, 2 cm and 4 cm fall below 0.3 dB, 0.34 dB and 0.52 dB, respectively. For the Mifare Ultralight tag at distances of 0 cm, 1 cm and 2 cm, 90% the errors fall below 0.12 dB, 0.16 dB and 0.35 dB, respectively. The mean errors of the two tags are only 0.29 dB and 0.1 dB, respectively. We can observe that Mifare Ultralight tag generally incurs smaller error than Mifare Classic tag. This may be because the Mifare Ultralight tag has a much smaller antenna size, which makes it more sensitive to the field strength. As a result, it has a narrower transition region, which conforms the finding in Section 7.8.3. This makes Mifare Ultralight tag more responsive to our adaptive attenuation algorithm, resulting in a smaller estimation error.

Next we evaluate the eavesdropping distances achieved with our sniffer at different initiator-tag distances. We record the eavesdropping distances at which the received signal strength of the initiator falls below 100 mV by following the same procedure of the measurement study in Section 7.3. The results are summarized in Table 7.2. It can be seen that, for each tag, the eavesdropping distance decreases with the initiator-tag distance. This is because a longer initiator-tag distance requires a stronger signal strength to ensure reliable communication, which increases the eavesdropping distance. We also notice that the Mifare Ultralight tag always incurs longer eavesdropping distances than Mifare Classic tag. This is because the low-sensitivity receiver of the Mifare Ultralight tag requires higher transmission power to maintain reliable communication. The shortest eavesdropping distances for the two tags are 48 cm and 70 cm, respectively. It is worth noting that, even after significant reduction, the resulted eavesdropping distance may still be further than

	Initiator-tag Distance			
	0 cm	1 cm	2 cm	4 cm
Classic	48 cm	75 cm	110 cm	140 cm
Ultralight	70 cm	92 cm	122 cm	151 cm

Table 7.2: Eavesdropping distances after attenuation.

the expected NFC working distance. This is largely due to the fundamental design trade-off of NFC. nShield could apply higher attenuation to decrease the eavesdropping distance to only a few centimeters, but this would significantly reduce the reliability of the NFC communication.

7.9 Discussion

Although NFC does not support single-initiator-multiple-target communication, the presence of multiple target devices may lead to collisions in the discovery process. NFC standards require the initiator to resolve collisions observed in discovery process using anti-collision techniques similar to RFID standards, and interact with resolved targets one by one after the discovery process. nShield currently does not consider the multiple tag case. However, nShield can learn if a collision has occurred by overhearing the traffic from the initiator, and act accordingly. However, this extension is left for future work.

nShield significantly improves the amount of harvested energy over existing NFC-based energy harvesting systems [50][26][44][19]. However, compared to specialized wireless power transfer systems [65] that often achieve power harvesting efficiencies of at least 70%, nShield's efficiency is much lower (24.4%). This is mainly because the current NFC initiator is not optimized for high efficiency wireless power transfer. The antenna on NFC transmitter has low Q-factor, which significantly limits the power transfer efficiency. Moreover, achieving high efficiency also requires that the transmitter and receiver must be precisely tuned to the same resonant frequency, which varies with the transmitter-receiver distance. High efficiency inductive power transfer systems adopt several techniques including resonant frequency auto-tuning and antenna impedance auto-tuning to deal with the detuning effects. Unfortunately, these mechanisms are not implemented on

NFC initiators.

We acknowledge that a complete redesign of the NFC initiator would be a more effective way to improve physical security. However, such a "clean-slate" approach may prove challenging in practice due to the need of involving many players (from IC to device manufacturers). Moreover, this would leave the legacy devices already shipped exposed to malicious attacks. The next-generation NFC chipsets may have native transmission control capabilities, which allow mobile devices to configure their NFC transmission power from software. This eliminates the need of accessory security hardware like nShield. In such a case, the adaptive attenuation algorithm of nShield can be integrated by the NFC driver to attenuate the transmission power.

Thanks to the high energy harvesting efficiency, the nShield platform is capable of powering additional hardware components like sensors. Moreover, it can be used as a software-defined radio platform for studying NFC protocols.

CHAPTER 8

CONCLUSION

In the last decade, we have witnessed the increasing adoption of wireless technologies like WiFi, Cellular, Bluetooth, ZigBee, and NFC. However, the fast growth of wireless networks generates significant interference, which leads to network performance degradation and security issues. In this dissertation, we utilize novel physical layer techniques to deal with the interference, which improve the performance and security of sensor networks and NFC systems, respectively.

First, we exploit the WiFi interference as a “blessing" for sensor networks and explore several approaches to utilize such interference, which are summarized as follows.

1. We propose new WiFi interference detection techniques. A new DSP algorithm called Common Multiple Folding (CMF) is developed to amplify signals with unknown periods in WiFi interference samples. We also adopt a constant false alarm rate (CFAR) detector that can minimize the false negative (FN) rate of WiFi AP detection while satisfying the user-specified upper bound on false positive (FP) rate.

2. We develop a system called *ZiFi* that utilizes ZigBee radio to identify the existence of WiFi networks. We evaluate ZiFi on two platforms, Linux netbook connected to a TelosB mote through the USB interface, and Nokia N73 smartphone that integrates a ZigBee card through the miniSD interface. Our results show that ZiFi can detect WiFi APs with high accuracy, short delay, and little overhead.

3. We propose a ZigBee-based WLAN monitoring system called WizNet. Powered by batteries, WizNet nodes can be deployed in large quantities to monitor the spatial performance of a WLAN in long periods of time. By adopting digital signal processing techniques, WizNet automatically identifies 802.11 signals from ZigBee RSS measurements, associates them with wireless access points, and accurately estimate the SNR and channel utilization rate.

WizNet can also collect user statistics based on RSS signatures of 802.11 access point scans and discover rogue APs. WizNet has been implemented in TinyOS 2.x and extensively evaluated on a wireless testbed consisting of 26 TelosB motes and 802.11 nodes. Our results over a period of 140 hours show that WizNet can accurately capture the spatial and temporal performance variability of a large-scale production WLAN.

4. We proposes a novel WSN time synchronization approach that enables ZigBee node to detect and synchronize to the periodic beacons broadcasted by WiFi APs. We experimentally characterized the characteristics of WiFi beacons in an enterprise WiFi network consisting of over 50 APs deployed in a 300,000 square foot office building. We implement a sensor network time synchronization protocol called WizSync in TinyOS 2.1x, which exploits such periodic WiFi beacon interference as reference clock. We conduct extensive evaluation on a testbed consisting of 19 TelosB motes. Our results show that WizSync can achieve similar synchronization accuracy as FTSP while incurring only a fraction of its power consumption. In a 10-day experimental evaluation, WizSync achieved an average synchronization error of 0.12 milliseconds with per-node power consumption of 50.9 *uW*.

Second, we propose a novel, noninvasive NFC security system called *nShield* to reduce the amount of interference signals generated by NFC radios, which protects NFC against passive eavesdropping. nShield is a credit card-sized thin pad that can be easily stuck on the back of mobile devices (see Fig. 7.6). nShield implements a novel adaptive RF attenuation scheme, in which the extra RF energy of NFC transmissions is determined and absorbed by nShield. At the same time, nShield scavenges the extra RF energy to sustain the perpetual operation. A key contribution of this work is the analysis of the factors affecting the energy harvesting efficiency, and the design of a highly effective energy harvesting system. nSheild is capable of harvesting significant amount of power (55 mW) from commodity mobile devices, which is at least a 1.8X improvement over the state-of-the-art NFC-based energy harvesting systems. Together with the extremely lo-power design, it enables nShield to provide the host uninterrupted protection against malicious

eavesdropping. Lastly, the small form factor, self-sustainability, and transparency to OS, makes nShield an attractive solution to retrofit existing mobile devices with protection against passive eavesdropping.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Texas Instruments Inc., BRF6300 BlueLink 5.0.

[2] Adafruit pn532 breakboard. http://www.adafruit.com/products/364.

[3] Airtight netwoks. http://airtightnetworks.net.

[4] Americans spend 58 mins a day on smartphones. http://www.experian.com/blogs/marketing-forward/2013/05/28/americans-spend-58-minutes-a-day-on-their-smartphones/.

[5] Berkeley wifi products. http://www.bvsystems.com/Products/WLAN/WLAN.htm.

[6] Crawdad: A community resource for archiving wireless data at dartmouth. http://crawdad.cs.dartmouth.edu/.

[7] D-itg distributed internet traffic generator. http://www.grid.unina.it/software/ITG/.

[8] Des wikipedia site. http://en.wikipedia.org/wiki/Data_Encryption_Standard.

[9] Fluke airmagnet wifi measurement tools. http://www.airmagnet.com/solutions/.

[10] Hobbes & co., ltd., wl-f601pro digital wifi detector. http://www.hobbes-europe.com/product.php5?products_id=79.

[11] How soon is now: Nfc smartphones and physical access control systems. http://blogs.gartner.com/mark-diodati/2011/10/31/how-soon-is-now-nfc-smartphones-and-physical-access-control-systems/.

[12] Impedance matching wikipedia site. http://en.wikipedia.org/wiki/Impedance_matching.

[13] Iperf. http://iperf.sourceforge.net/.

[14] Metageek, llc, wi-spy spectrum analyzer. http://www.metageek.net/.

[15] Mobile payments today. http://www.mobilepaymentstoday.com/research/400/Contactless-NFC.

[16] Near field communication (nfc) 2014-2024. http://www.prnewswire.com/news-releases/near-field-communication-nfc-2014-2024-227654461.html.

[17] Near field communication wikipedia. http://en.wikipedia.org/wiki/Near_field_communication.

[18] Nfc forum technical specifications. http://www.nfc-forum.org/specs/spec_list/.

[19] Nfc-wisp project site. http://www.alansonsample.com/research/NFC-WISP.html.

[20] Nxp: Pn532 user manual. http://www.nxp.com/documents/user_manual/141520.pdf.

[21] Pantech & curitel p1 cell phone. http://www.curitel.com.

[22] Proxmark 3 project site. http://www.proxmark.org/.

[23] Rf micro devices inc. bluetooth transceiver rf2968. http://www.rfmd.com/.

[24] Rsa wikipedia site. http://en.wikipedia.org/wiki/RSA_%28cryptosystem%29.

[25] Sigcomm 2008 traces. http://www.cs.umd.edu/projects/wifidelity/sigcomm08_traces/.

[26] St discovery kit. http://www.st.com/web/en/catalog/tools/FM116/SC1444/PF253360.

[27] Still not a wallet, nfc has a second life as a safe, simple pairing tool. http://gigaom.com/2013/08/08/still-not-a-wallet-nfc-has-a-second-life-as-a-safe-simple-pairing-tool/.

[28] Strasbourg nfc ticketing moves to commercial launch. http://www.nfcworld.com/2013/07/05/324901/strasbourg-nfc-ticketing-moves-to-commercial-launch/.

[29] Ti:hf antenna design notes. http://www.ti.com/rfid/docs/manuals/appNotes/HFAntennaDesignNotes.pdf.

[30] Wep wikipedia site. http://en.wikipedia.org/wiki/Wired_Equivalent_Privacy.

[31] Wifi hotspots set to more than triple by 2015. http://www.informa.com/Media-centre/Press-releases--news/Latest-News/Wifi-hotspots-set-to-more-than-triple-by-2015/.

[32] Zigbee alliance, products & certification overview. http://www.zigbee.org/Products/Overview.aspx.

[33] Y. Agarwal, R. Chandra, A. Wolman, P. Bahl, K. Chin, and R. Gupta. Wireless wakeups revisited: energy management for voip over wi-fi smartphones. In *Proceedings of the 5th international conference on Mobile systems, applications and services*, pages 179–191. ACM, 2007.

[34] Y. Agarwal, C. Schurgers, and R. Gupta. Dynamic power management using on demand paging for networked embedded systems. In *Proceedings of the 2005 Asia and South Pacific Design Automation Conference*, pages 755–759. ACM, 2005.

[35] G. Ananthanarayanan and I. Stoica. Blue-fi: enhancing wi-fi performance using bluetooth signals. In *Proceedings of the 7th international conference on Mobile systems, applications, and services*, pages 249–262. ACM, 2009.

[36] C. A. Boano, T. Voigt, C. Noda, K. Romer, and M. Zúñiga. Jamlab: Augmenting sensornet testbeds with realistic and controlled interference generation. In *Information Processing in Sensor Networks (IPSN), 2011 10th International Conference on*, pages 175–186. IEEE, 2011.

[37] V. Bychkovsky, B. Hull, A. Miu, H. Balakrishnan, and S. Madden. A measurement study of vehicular internet access using in situ wi-fi networks. In *Proceedings of the 12th annual international conference on Mobile computing and networking*, pages 50–61. ACM, 2006.

[38] R. Chandra, J. Padhye, A. Wolman, and B. Zill. A location-based management system for enterprise wireless lans. In *NSDI*, 2007.

[39] R. Chandra, V. N. Padmanabhan, and M. Zhang. Wifiprofiler: cooperative diagnosis in wireless lans. In *Proceedings of the 4th international conference on Mobile systems, applications and services*, pages 205–219. ACM, 2006.

[40] K. Chebrolu and A. Dhekne. Esense: communication through energy sensing. In *Proceedings of the 15th annual international conference on Mobile computing and networking*, pages 85–96. ACM, 2009.

[41] Y. Chen, Q. Wang, M. Chang, and A. Terzis. Ultra-low power time synchronization using passive radio receivers. In *Information Processing in Sensor Networks (IPSN), 2011 10th International Conference on*, pages 235–245. IEEE, 2011.

[42] Y.-C. Cheng, M. Afanasyev, P. Verkaik, P. Benkö, J. Chiang, A. C. Snoeren, S. Savage, and G. M. Voelker. *Automating cross-layer diagnosis of enterprise wireless networks*, volume 37. ACM, 2007.

[43] A. Dementyev, J. Gummeson, D. Thrasher, A. Parks, D. Ganesan, J. R. Smith, and A. P. Sample. Wirelessly powered bistable display tags. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*. ACM, 2013.

[44] J. Elson, L. Girod, and D. Estrin. Fine-grained network time synchronization using reference broadcasts. *ACM SIGOPS Operating Systems Review*, 36(SI):147–163, 2002.

[45] H. Falaki, R. Mahajan, S. Kandula, D. Lymberopoulos, R. Govindan, and D. Estrin. Diversity in smartphone usage. In *Proceedings of the 8th international conference on Mobile systems, applications, and services*. ACM, 2010.

[46] S. Ganeriwal, R. Kumar, and M. B. Srivastava. Timing-sync protocol for sensor networks. In *Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 138–149. ACM, 2003.

[47] R. K. Ganti, P. Jayachandran, H. Luo, and T. F. Abdelzaher. Datalink streaming in wireless sensor networks. In *Proceedings of the 4th international conference on Embedded networked sensor systems*, pages 209–222. ACM, 2006.

[48] S. Geirhofer, L. Tong, and B. M. Sadler. Dynamic spectrum access in wlan channels: Empirical model and its stochastic analysis. In *Proceedings of the first international workshop on Technology and policy for accessing spectrum*, page 14. ACM, 2006.

[49] J. J. Gummeson, B. Priyantha, D. Ganesan, D. Thrasher, and P. Zhang. Engarde: Protecting the mobile phone from malicious nfc interactions. In *Proceeding of the 11th annual international conference on Mobile systems, applications, and services*. ACM, 2013.

[50] G. Hancke. Practical attacks on proximity identification systems. In *Security and Privacy, 2006 IEEE Symposium on*, pages 6 pp.–333, 2006.

[51] T. Hao, R. Zhou, G. Xing, and M. Mutka. Wizsync: Exploiting wi-fi infrastructure for clock synchronization in wireless sensor networks. In *Real-Time Systems Symposium (RTSS), 2011 IEEE 32nd*, pages 149–158. IEEE, 2011.

[52] E. Haselsteiner and K. Breitfu? Security in near field communication (nfc). In *Printed handout of Workshop on RFID Security, July 2006.*

[53] T. He, S. Krishnamurthy, J. A. Stankovic, T. Abdelzaher, L. Luo, R. Stoleru, T. Yan, L. Gu, J. Hui, and B. Krogh. Energy-efficient surveillance system using wireless sensor networks. In *Proceedings of the 2nd international conference on Mobile systems, applications, and services*, MobiSys '04, pages 270–283, New York, NY, USA, 2004. ACM.

[54] J. Huang, G. Xing, G. Zhou, and R. Zhou. Beyond co-existence: Exploiting wifi white space for zigbee performance assurance. In *Network Protocols (ICNP), 2010 18th IEEE International Conference on*, pages 305–314. IEEE, 2010.

[55] K. Jamieson and H. Balakrishnan. Ppr: Partial packet recovery for wireless networks. *ACM SIGCOMM Computer Communication Review*, 37(4):409–420, 2007.

[56] J. J. Karakash. *Transmission lines and filter networks*. Macmillan New York, 1950.

[57] M. Kesler. Highly resonant wireless power transfer: Safe, efficient, and over distance. *WiTricity Corporation, Watertown, MA, USA*, 2013.

[58] K.-H. Kim, A. W. Min, and K. G. Shin. Sybot: an adaptive and mobile spectrum survey system for wifi networks. In *Proceedings of the sixteenth annual international conference on Mobile computing and networking*, pages 293–304. ACM, 2010.

[59] J. Koo, R. K. Panta, S. Bagchi, and L. Montestruque. A tale of two synchronizing clocks. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, pages 239–252. ACM, 2009.

[60] H. S. Kortvedt and S. F. Mj?lsnes. Eavesdropping near field communication. In *The Norwegian Information Security Conference (NISK) 2009*.

[61] A. Kurs, A. Karalis, R. Moffatt, J. D. Joannopoulos, P. Fisher, and M. Soljačić. Wireless power transfer via strongly coupled magnetic resonances. *science*, 317(5834):83–86, 2007.

[62] L. Li, G. Xing, L. Sun, W. Huangfu, R. Zhou, and H. Zhu. Exploiting fm radio data system for adaptive clock calibration in sensor networks. In *Proceedings of the 9th international conference on Mobile systems, applications, and services*, pages 169–182. ACM, 2011.

[63] J.-P. M. G. Linmartz. *Wireless Communication, The Interactive Multimedia CD-ROM*. Baltzer Science Publishers, 1996.

[64] Z. N. Low, R. Chinga, R. Tseng, and J. Lin. Design and test of a high-power high-efficiency loosely coupled planar wireless power transfer system. *Industrial Electronics, IEEE Transactions on*, 56(5):1801–1812, 2009.

[65] R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan. Analyzing the mac-level behavior of wireless networks in the wild. In *ACM SIGCOMM Computer Communication Review*, volume 36, pages 75–86. ACM, 2006.

[66] M. Maróti, B. Kusy, G. Simon, and Á. Lédeczi. The flooding time synchronization protocol. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 39–49. ACM, 2004.

[67] C. Miller. Exploring the nfc attack surface. *Proceedings of Blackhat*, 2012.

[68] N. Mishra, K. Chebrolu, B. Raman, and A. Pathak. Wake-on-wlan. In *Proceedings of the 15th international conference on World Wide Web*, pages 761–769. ACM, 2006.

[69] N. Mishra, D. Golcha, A. Bhadauria, B. Raman, and K. Chebrolu. S-wow: Signature based wake-on-wlan. In *Communication Systems Software and Middleware, 2007. COMSWARE 2007. 2nd International Conference on*, pages 1–8. IEEE, 2007.

[70] A. J. Nicholson and B. D. Noble. Breadcrumbs: forecasting mobile connectivity. In *Proceedings of the 14th ACM international conference on Mobile computing and networking*, pages 46–57. ACM, 2008.

[71] T. Pering, Y. Agarwal, R. Gupta, and R. Want. Coolspots: reducing the power consumption of wireless mobile devices with multiple radio interfaces. In *Proceedings of the 4th international conference on Mobile systems, applications and services*, pages 220–232. ACM, 2006.

[72] J. Polastre, R. Szewczyk, and D. Culler. Telos: enabling ultra-low power wireless research. In *Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on*, pages 364–369. IEEE, 2005.

[73] A. Rahmati and L. Zhong. Context-for-wireless: context-sensitive energy-efficient wireless

data transfer. In *Proceedings of the 5th international conference on Mobile systems, applications and services*, pages 165–178. ACM, 2007.

[74] M. R. Rieback, G. Gaydadjiev, B. Crispo, R. F. Hofman, and A. S. Tanenbaum. A platform for rfid security and privacy administration. In *USENIX LISA*, pages 89–102, 2006.

[75] J. Robinson, R. Swaminathan, and E. W. Knightly. Assessment of urban-scale wireless networks with a small number of measurements. In *Proceedings of the 14th ACM international conference on Mobile computing and networking*, pages 187–198. ACM, 2008.

[76] I. Rose and M. Welsh. Mapping the urban wireless landscape with argos. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, pages 323–336. ACM, 2010.

[77] A. Rowe, V. Gupta, and R. R. Rajkumar. Low-power clock synchronization using electromagnetic energy radiating from ac power lines. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, pages 211–224. ACM, 2009.

[78] A. Rowe, R. Mangharam, and R. Rajkumar. RT-Link: A global time-synchronized link protocol for sensor networks. *Ad Hoc Networks*, 6(8):1201–1220, 2008.

[79] E. R.V.E.LOVELACE, J.M.SUTTON. Digital search methods for pulsars. In *Nature.Vol.222*, 1969.

[80] T. Schmid, P. Dutta, and M. B. Srivastava. High-resolution, low-power time synchronization an oxymoron no more. In *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, pages 151–161. ACM, 2010.

[81] E. Shih, P. Bahl, and M. J. Sinclair. Wake on wireless: an event driven energy saving strategy for battery operated devices. In *Proceedings of the 8th annual international conference on Mobile computing and networking*, pages 160–171. ACM, 2002.

[82] J. Sorber, N. Banerjee, M. D. Corner, and S. Rollins. Turducken: hierarchical power management for mobile devices. In *Proceedings of the 3rd international conference on Mobile systems, applications, and services*, pages 261–274. ACM, 2005.

[83] D. H. Staelin. Fast folding algorithm for detection of periodic pulse trains. In *IEEE Proceedings*, volume 57, pages 724–725, 1969.

[84] K. K. Talukdar and W. D. Lawing. Estimation of the parameters of the rice distribution. *the Journal of the Acoustical Society of America*, 89(3):1193–1197, 1991.

[85] F. Vanheel, J. Verhaevert, and I. Moerman. Study on distance of interference sources on wireless sensor network. In *Microwave Conference, 2008. EuMC 2008. 38th European*, pages 175–178. IEEE, 2008.

[86] P. Varshney. *Distributed Detection and Data Fusion*. Spinger-Verlag, New York, NY, 1996.

[87] D. Welch and S. Lathrop. Wireless security threat taxonomy. In *Information Assurance Workshop, 2003. IEEE Systems, Man and Cybernetics Society*, pages 76–83, 2003.

[88] Y. Z. Wenxian Li and T. He. Wibee: Building wi-fi radio map with zigbee sensor networks. In *IEEE INFOCOM '12 Mini-Conference, 2012*.

[89] J. Wilson, V. Bhargava, A. Redfern, and P. Wright. A wireless sensor network and incident command interface for urban firefighting. In *Mobile and Ubiquitous Systems: Networking Services, 2007. MobiQuitous 2007. Fourth Annual International Conference on*, pages 1–7, 2007.

[90] Y. Wu, G. Zhou, and J. A. Stankovic. Acr: Active collision recovery in dense wireless sensor networks. In *INFOCOM, 2010 Proceedings IEEE*, pages 1–9. IEEE, 2010.

[91] G. Xing, M. Sha, J. Huang, G. Zhou, X. Wang, and S. Liu. Multi-channel interference measurement and modeling in low-power wireless networks. In *Real-Time Systems Symposium, 2009, RTSS 2009. 30th IEEE*, pages 248–257. IEEE, 2009.

[92] Y. Xiong, R. Zhou, M. Li, J. Ma, L. Sun, and G. Xing. Zifi: Exploiting cross-technology interference signatures for wireless lan discovery. *IEEE Transactions on Mobile Computing*, 99(PrePrints):1, 2014.

[93] J. Yeo, M. Youssef, T. Henderson, and A. Agrawala. An accurate technique for measuring the wireless side of wireless networks. In *Papers presented at the 2005 workshop on Wireless traffic measurements and modeling*, pages 13–18. USENIX Association, 2005.

[94] X. Zhang, X. Wang, Y. Ren, C. Chen, and J. Ma. A novel compatible hardware expansion method based on general memory interface. In *Communications and Mobile Computing, 2009. CMC'09. WRI International Conference on*, volume 2, pages 601–605. IEEE, 2009.

[95] R. Zhou and G. Xing. nshield: A noninvasive nfc security system for mobile devices. In *Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '14, 2014.

[96] R. Zhou, G. Xing, X. Xu, J. Wang, and L. Gu. Wiznet: A zigbee-based sensor system for distributed wireless lan performance monitoring. In *Pervasive Computing and Communications (PerCom), 2013 IEEE International Conference on*, March 2013.