Embracing Interference in Wireless Systems

ACM Books

Embracing Interference in Wireless Systems

Shyamnath Gollakota
University of Washington

ACM Books #1

Embracing Interference in Wireless Systems

Shyamnath Gollakota

books.acm.org
www.morganclaypool.com

ISBN: 978-1-62705-474-4 print
ISBN: 978-1-62705-475-1 ebook
ISBN: 978-1-62705-476-8 ePub
Series ISSN: (to come)

DOIs: 10.1145/2611390 Book
10.1145/2611390.2611391 Preface
10.1145/2611390.2611392 Chapter 1
10.1145/2611390.2611393 Chapter 2
10.1145/2611390.2611394 Chapter 3
10.1145/2611390.2611395 Chapter 4
10.1145/2611390.2611396 Chapter 5
10.1145/2611390.2611397 Chapter 6
10.1145/2611390.2611398 Bibliography

A publication in the ACM Books series, #1
Editor in Chief: M. Tamer Özsu, *University of Waterloo*

First Edition

10 9 8 7 6 5 4 3 2 1

To Rajeswari and Subramanyam Gollakota

Contents

Preface

The wireless medium is a shared resource. If nearby devices transmit at the same time, their signals interfere, resulting in a collision. In traditional networks, collisions cause the loss of the transmitted information. For this reason, wireless networks have been designed with the assumption that interference is intrinsically harmful and must be avoided.

This book takes an alternate approach: instead of viewing interference as an inherently counterproductive phenomenon that should to be avoided, we design practical systems that transform interference into a harmless, and even beneficial, phenomenon.

To achieve this goal, we consider how wireless signals interact when they interfere, and use this understanding in our system designs. Specifically, when interference occurs, the signals get mixed on the wireless medium. By understanding the parameters of this mixing, we can invert the mixing and decode the interfered packets, thus making interference harmless. Furthermore, we can control this mixing process to create strategic interference that allow decodability at a particular receiver of interest, but prevent decodability at unintended receivers and adversaries. Hence, we can transform interference into a beneficial phenomenon that provides security.

Building on this approach, we make four main contributions. First, we present the first WiFi receiver that can successfully reconstruct the transmitted information in the presence of packet collisions. Next, we introduce a WiFi receiver design that can decode in the presence of high-power cross-technology interference from devices like baby monitors, cordless phones, microwave ovens, or even unknown technologies. We then show how we can harness interference to improve security. In particular, we develop the first system that secures an insecure medical implant without any modification to the implant itself. Finally, we present a solution that establishes secure connections between any two WiFi devices, without having users enter passwords or use pre-shared secret keys.

Acknowledgments

The work presented in this book was done over six years of my graduate studies with my advisor, Dina Katabi. From my first year, despite my unfamiliarity with research, she spent hours daily discussing ideas, encouraging me to push the boundaries, and providing encouragement and support. The best times I had at MIT were the endless arguments about ideas I had with her, and in the process coming up with even better ideas. I am deeply indebted to Dina and hope to be as good an advisor when nurturing my own students.

I also had a wonderful experience working with Nickolai Zeldovich. We spent hours together, building upon each other's expertise to come up with the ideas in our Usenix Security paper on TEP. I also thank him for sitting through countless practice talks and helping with my job talk. I would also like to thank my thesis committee: Hari Balakrishnan, Muriel Medard, Srini Seshan, and Kevin Fu. I learned a lot from them, and my thesis has benefited significantly from their experience and feedback.

Much of the work in this book was possible because of a fantastic group of student collaborators: Fadel Adib, Haitham Hassaneih, Ben Ransford, and Nabeel Ahmed. As an undergraduate interning over the summer, Fadel contributed significantly to our SIGCOMM paper on TIMO. Without any prior experience with signals or wireless networks, Fadel implemented the basic framework for TIMO within just two months. Haitham was instrumental in getting our SIGCOMM paper on IMDShield submitted in a near perfect state. Likewise, Ben's writing significantly helped the IMDShield paper. Nabeel collaborated on our Usenix Security paper on TEP; his mastery of the WiFi driver helped us implement TEP on WiFi cards in less than three weeks.

I had the great pleasure of sharing my journey through graduate school with Sachin Katti, Nate Kushman, Srikanth Kandula, Hariharan Rahul, Swarun Kumar, Jue Wang, Ali Shoeb, Mythili Vutukuru, Lenin Sivalingam, Kyle Jamieson, Szymon Jakubczak, Katrina LaCurts, Arvind Thiagarajan, Raluca Ada Popa, Ramesh Chandra, Micah Brodsky, Samuel Perli, Dorothy Curtis, and Mary McDavit.

Sometimes it is easier to lose track of the bigger picture and the life outside the lab. Shounak Simlai kept me grounded and made sure that I spent enough time outside CSAIL. He also made me question every decision I made, allowing me to emerge stronger and more decisive. I am greatly indebted to him for all of it. I would also like to express my earnest gratitude to my parents, my brother, and my grandmother, all of whom made countless sacrifices to raise me and to give me the best possible education. They have always been a rock behind me, encouraging, loving, and supporting me unconditionally. This book is dedicated to them.

1 Introduction

Wireless and mobile systems play an increasingly important role in our lives. Studies show that mobile traffic will increase by multiple orders of magnitude over the next few years [Rysavy Research 2010]. This increase is driven by user demands for mobile video and their desire to access data from hand-held devices. It is also emphasized by the incorporation of wireless communication in most modern applications, including public safety, vehicular networks, home networks, and health care. In fact, nowadays even medical implants are equipped with wireless connectivity and rely on wireless communication to send data and receive commands.

Wireless systems, however, are severely limited by the phenomenon of interference. The wireless medium is a shared resource. If nearby devices transmit at the same time, their signals interfere, resulting in a collision. Traditionally, collisions have been considered to cause the loss of the transmitted information. For this reason, practical wireless systems have been designed with the assumption that interference is intrinsically harmful, and must be avoided. Completely avoiding interference, however, is infeasible, particularly for systems operating in the unlicensed spectrum like WiFi and sensors [Cisco 2010, Akyildiz et al. 2002]. Hence, these technologies try to hide interference underneath a logical interface called a "packet". Packets that suffer from interference are considered lost, and the system processes only interference-free packets. While simple, this all-or-nothing packet-based interface becomes inefficient as the network scales to a large number of users and evolves to support new applications. The more devices that try to access the medium and the larger the diversity of their technologies, the higher the chances of interference and the less effective today's systems are at hiding it [Verdu 1998, Tse and Vishwanath 2005, Farpoint Group 2010, Bandspeed 2010].

1.1 Embracing Wireless Interference

In this book, we take an alternate approach: instead of hiding interference, we embrace it, i.e., we aim at understanding how interfering wireless signals interact over the wireless medium and exploit this understanding in our designs. We are motivated by theoretical results in information theory which show that interference does not necessarily cause the loss of information [Gallager 1985, Tse and Vishwanath 2005]; when interference happens, the signals get mixed on the wireless medium. By understanding the parameters of this mixing in practical systems over real wireless channels, this book develops systems that can invert the mixing and decode the interfered packets, hence rendering interference harmless. Furthermore, it also shows how to manipulate the interfering signals by controlling the parameters of the mixing, so that we create strategic interference that allows decodability at a particular receiver of interest, but prevents decodability at unintended receivers and adversaries. Hence, we can transform interference into a beneficial phenomenon that provides security.

The book introduces a family of algorithms and system architectures for treating interference in practical wireless networks. Our designs have three key properties that make them practical. First, they do not assume global knowledge of the network and can operate in a fully distributed manner. Second, they take into account realistic limitations such as radio frequency offset, the lack of very tight signal synchronization, feedback overhead, bursty traffic, etc. Third, they are all implemented and empirically evaluations in wireless testbeds to demonstrate their feasibility and high performance.

Our systems do not limit themselves by the traditional packet-based interface to the wireless medium; they take an integrated approach that jointly optimizes signal transmission over the wireless medium and packet-based network protocols. Empirical results from our systems show that such an integrated approach delivers large performance and security gains.

The systems developed in this book address the following four challenges that face today's wireless systems, contributing to both wireless performance and security.

Decoding collisions. ZigZag is the first receiver design that decodes WiFi collisions without sender modification and without any assumptions of packet synchronization, large differences in power, or special codes. This work exploits the natural asynchrony between colliding transmitters which results in stretches of interference-free bits at the start of a collision; it uses these interference-free bits to infer important parameters regarding the structure of the collision which it exploits to bootstrap the decoder.

Combating high-power cross-technology interference. The unlicensed spectrum is increasingly crowded with a multitude of diverse technologies that interfere with WiFi and cause loss of connectivity. We have designed and built TIMO, the first WiFi receiver that can decode in the presence of high-power cross-technology interference. TIMO is agnostic to the interferer's technology and hence allows us to have one solution to deal with interference from diverse technologies. We also introduce a new form of cognitive communication where different technologies do not necessarily have to use isolated frequencies, as in traditional cognitive communication, but could in crowded environments use the same frequency band. This enables packing more radios and data into the wireless spectrum.

Securing wireless medical implants without access to the implant itself. Today, millions of people have medical implants (e.g., pacemaker) with wireless connectivity. Past work has shown that these devices are susceptible to attacks over the wireless channel that can change their therapy or listen to their private data. Medical implants, however, have a lifetime of about ten years. Fixing the security problem in the millions of implanted devices will require the patients to go through difficult surgery. We present the first system that provides confidentiality for implants' transmissions and protects them against commands from unauthorized parties, without requiring any modification to the implants themselves. Further, because our design provides protection at the signal level, it can also be used as a complementary defense-in-depth solution to devices that feature cryptographic or other application-layer protection mechanisms.

Making security easy for ordinary users. Last, we introduce the first system that enables WiFi users to establish secure connections without any passwords, prior key distribution, or out-of-band channels. The main idea is to exploit that once a random wireless signal is transmitted, an attacker cannot eliminate the energy that the signal produces on the wireless medium. Hence, the attacker cannot hide the fact that the signal was transmitted. Leveraging this simple idea, we construct a new secure message type that can neither be altered nor hidden without detection. We analytically prove the security of the resulting protocol and empirically demonstrate its practicality.

Before we dig into the details of these systems, we provide the high-level idea underlying each of them and describe their relation to prior work.

1.1.1 Decoding 802.11 Collisions

ZigZag, the subject of Chapter 2, introduces the first WiFi receiver that decodes packet collisions. Collisions are a known problem in WiFi networks [Cheng et al. 2006, Khurana et al. 1998, Judd and Steenkiste 2005, Ng et al. 2005, Ware et al. 2000]. Current WiFi networks attempt to counter collisions by making senders sense the medium, and abstain from transmitting when the medium is busy. This approach, however, fails in multiple cases, including the classic hidden terminal problem [Karn 1990, Bharghavan et al. 1994]. Specifically, consider the scenario in Figure 1.1, where two transmitters,

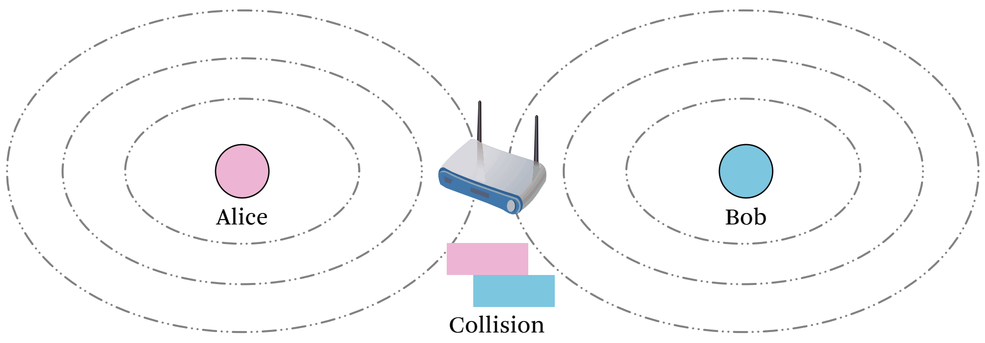


Figure 1.1 **Wireless Collisions.** When two wireless devices, Alice and Bob, transmit at the same time, their packets interfere with each other, resulting in packet collisions.

Alice and Bob, want to communicate with the access point (AP). Yet, due to some obstruction (e.g., a wall) they cannot sense each other's signals. As a result, they end up transmitting at the same time, causing the packets to collide at the access point and become undecodable. Since neither node receives an acknowledgement, Alice and Bob retransmit the same packets with a small random jitter. The small jitter is originally intended to enable one of the two transmitters to start first, which in the absence of hidden terminal effects causes the other transmitter to sense the ongoing transmission and abstain from transmitting. However, in the hidden terminal scenario, since the two transmitters do not hear each other, Alice and Bob will continue increasing the jitter and retransmitting their packets, until eventually they get a packet through or they time out. Either way, both transmitters experience a loss rate that is too high for useful applications [Judd and Steenkiste 2005, Ware et al. 2000].

In contrast to all prior work on hidden terminals which aims to avoid collisions [Karn 1990, Bharghavan et al. 1994, Netgear 2005], we ask the following question: Can we design an access point that takes two collisions—such as the collision of Alice's and Bob's packets and that of their retransmissions—and decodes the content of the two colliding packets? If we can design such an access point, we can deliver the two original packets, as if they were transmitted one after the other and no collision occurred.

The answer to this question is "yes." In particular, we developed ZigZag, a WiFi receiver that decodes collisions. ZigZag leverages that WiFi senders insert a small random delay before they start transmitting (the protocol requires them to pick a random slot from a window of 32 slots). Thus, if we consider the collision of Alice's and Bob's packets, and the collision resulting from their retransmission of the same two packets, it is likely that Bob's packet has a different offset in the two collisions, i.e., $\Delta_1 \neq \Delta_2$, as

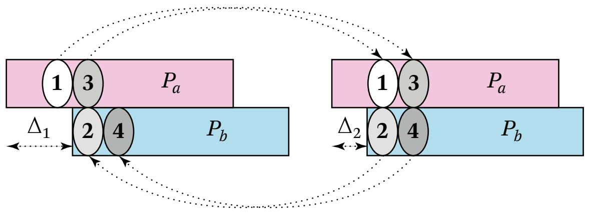


Figure 1.2 **ZigZag Decoding.** ZigZag first decodes chunk 1 in the first collision, which is interference-free. It subtracts chunk 1 from the second collision which causes chunk 2 to become interference-free and hence decodable. It then decodes chunk 2 and subtracts from the first collision causing chunk 3 to become interference-free and hence decodable. It proceeds like that until it has decoded both packets.

shown in Figure 1.2. Consequently, the access point (AP) can find a chunk of bits that is interference-free in one collision but experiences some interference in the other, such as chunk 1, in the figure. Since chunk 1 is interference-free in the first collision, the AP can decode it using a standard decoder. The AP then subtracts chunk 1 from the second collision to obtain chunk 2. Now, chunk 2 is interference-free, and hence can be decoded using a standard decoder. Next, the access point goes back to the first collision, subtracts chunk 2, which causes chunk 3 to become interference-free and hence decodable. A ZigZag access point proceeds in this manner decoding a chunk from one collision and subtracting it from the other until both packets are fully decoded. Thus, even in the face of collisions, ZigZag can correctly deliver two packets in two timeslots, as if the packets were transmitted sequentially with no collisions.

The above description is at a high level. Of course in practice we cannot subtract chunks of bits; we need to regenerate the signal associated with a chunk and apply the channel coefficients to it, before we subtract it from the received collision signal. Further, we need to estimate the offset values, Δ_1 and Δ_2, from the received signals. In Chapter 2, we describe how ZigZag addresses these practical issues and, further, generalize ZigZag to collisions between more than two devices.

Relation to Prior Work. ZigZag is the first WiFi receiver that decodes packet collisions. ZigZag builds on work in both networked systems and communication theory. Prior work in networked systems addresses the problem of collisions by trying to avoid them [Karn 1990, Bharghavan et al. 1994]. In contrast, ZigZag decodes packet collisions. Prior work in communication theory introduces the concept of interference cancellation [Halperin et al. 2008a, Hou et al. 2006]. However, traditional interference cancellation assumes that the transmitters know a priori that their packets will collide and coordinate their choice of code or transmission power accordingly. However, WiFi collisions typically happen because the transmitters cannot hear each other and hence cannot coordinate their transmissions. Thus, traditional interference cancellation cannot be used to decode WiFi collisions. ZigZag provides a new form of interference cancellation that does not require any such coordination and hence is suitable for fully distributed wireless networks, e.g., WiFi.

1.1.2 Combating High-Power Cross-Technology Interference

In the previous section, we presented ZigZag, a solution for decoding packets in the presence of WiFi interference. But, what about non-WiFi interference? Current WiFi networks suffer from strong cross-technology interference [Ofcom 2009]. Devices like baby monitors, microwave ovens, and cordless phones can cause WiFi networks to experience a complete loss of connectivity [Farpoint Group 2010]. According to Cisco, these devices are responsible for more than 50% of customer complaints with WiFi [Cisco 2010]. Today, there is no solution to enable WiFi to work in the frequencies used by these devices.

A natural question is whether we can use ZigZag to address interference from these devices. Intuitively, in ZigZag, we considered the interfered signal as a linear equation with two unknowns: Alice's packet and Bob's packet. With two such independent linear equations, one can solve for the two unknowns and decode the two packets. Since WiFi transmitters retransmit their packets, we had two equations in two unknowns and hence ZigZag was able to decode both packets. Unlike WiFi devices, non-WiFi devices typically do not retransmit. Thus, with cross-technology interference, we do not have two independent linear equations and hence we cannot decode in the presence of interference.

In Chapter 3, we present TIMO, the first WiFi receiver that can decode in the presence of high-power cross-technology interference. The basic idea underlying TIMO is simple: let us leverage the fact that most WiFi devices today come with multiple antennas. For example, a typical WiFi access point has about two or three antennas. These multi-antenna devices exhibit a well-known property: since the antennas are placed in different locations, a signal traverses different paths on the wireless medium and arrives at each antenna with a different channel (amplitude and phase). For example, as shown in Figure 1.3, the signal from the WiFi client traverses two different channels, h_1 and h_2, to reach the two antennas on the access point. Similarly, a baby monitor has channels, h_3 and h_4, to the access point's antennas. Thus, when the baby monitor interfers with the WiFi client, a two-antenna WiFi access point re-

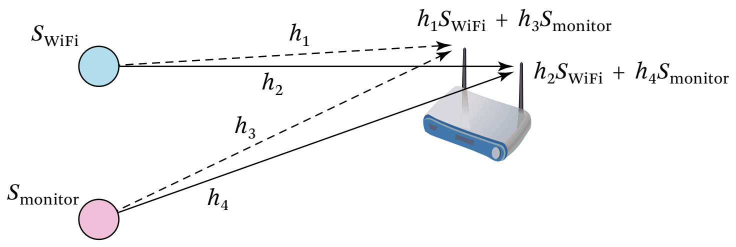

S_{WiFi}
h_1
$h_1S_{\text{WiFi}} + h_3S_{\text{monitor}}$
h_2
$h_2S_{\text{WiFi}} + h_4S_{\text{monitor}}$
h_3
S_{monitor}
h_4

Figure 1.3 **Dealing with cross-technology interference using multiple antennas.** The two antennas on the access point receive two independent linear equations which can be used to decode the WiFi signal in the presence of interference.

ceives two independent linear equations on its two antennas. Hence, if the access point can estimate the channels, it can solve the two linear equations and decode the WiFi signal.

The challenge, however, is that WiFi and baby monitors are diverse technologies that do not understand each other. As a result, a WiFi access point cannot compute a baby monitor's channel. In Chapter 3, we describe a gradient-descent style algorithm that enables us to decode the WiFi signal, without estimating the interferer's channel. Further, we show that our algorithm is oblivious to the technology of the interferer, i.e., it decodes the WiFi signal in the presence of interference from cordless phones, baby monitors, microwave ovens, or even an unknown technology.

Relation to Prior Work. TIMO is the first WiFi receiver that can decode in the presence of high-power cross-technology interference. TIMO builds on work in both networked systems and wireless communication. In network systems, TIMO builds on work in cognitive systems that avoid interference by having devices dynamically look for unused frequencies and using those frequencies [Chandra et al. 2008, Rahul et al. 2008]. In contrast, TIMO provides a new cognitive framework where diverse technologies coexist on the same frequency and hence can achieve much higher spectral efficiency. In wireless communication, TIMO builds on work in multi-antenna systems.[1] Traditional multi-antenna systems enable multiple transmitters of the same technology to transmit concurrently without interference. TIMO builds on this work and develops new algorithms that enable multi-antenna systems to work across diverse technologies.

1. http://www.arraycomm.com; last retrieved May 2014.

1.1.3 Non-Invasive Approach to Securing Medical Implants

Next, we harness interference to secure medical implants. Modern implantable medical devices (IMDs), including pacemakers, cardiac defibrillators, and neurostimulators, all feature wireless communication [Panescu 2008]. Adding wireless connectivity to IMDs has enabled remote monitoring of patients' vital signs and improved care providers' ability to deliver timely treatment, leading to a better health care system [Maisel 2005]. Recent work, however, has demonstrated that wireless connectivity can be exploited to compromise the confidentiality of IMDs' transmitted data or to send unauthorized commands to IMDs—even commands that cause the device to deliver an electric shock to the patient [Halperin et al. 2008b].

Traditionally, designers use cryptographic methods to provide confidentiality and prevent unauthorized access. However, adding cryptography *directly* to implants is difficult for two main reasons. First, millions of patients already have these unsecured implants [Zhan et al. 2007]. Once implanted, the device can last up to 10 years [Fu 2009]. It would be impractical for patients to undergo surgery to replace their implants with cryptography-enabled ones. Second, if the patient has an emergency and is taken to a foreign hospital where the doctor does not have the secret key, the implant cannot be accessed, which could be fatal.

In Chapter 4, we present IMDShield, the first solution that secures medical implants without modifying them. Our design delegates an implant's security to an external wearable device called the *shield*. Such an approach enhances the security of IMDs for patients who already have them, and empowers medical personnel to access a protected IMD by removing the external device or powering it off.

The shield protects the confidentiality of the IMD's transmitted data and also provides access control. Let us focus on the confidentiality question. Suppose an eavesdropper is snooping on the implant's data transmission. The shield monitors the medium, and whenever the implant transmits, it jams the implant's transmission with a random signal, as shown in Figure 1.4. Since the eavesdropper does not know the random interference signal, it cannot decode the implant's signal. However, simply creating interference and jamming not only prevents an eavesdropper from decoding the implant's transmissions but also prevents legitimate devices, including the shield, from receiving these transmissions.

Thus, the key question is this: How can the shield use interference to prevent adversaries from decoding the implant's signal, yet still be able to decode that signal and forward it to the doctor? At a high level, the received signal is a linear combination of the shield's random interference signal and the implant's signal. Since the shield generated the random interference signal, in principle, it can subtract this signal and decode the implant's transmissions. To do so, however, the shield needs to be able to

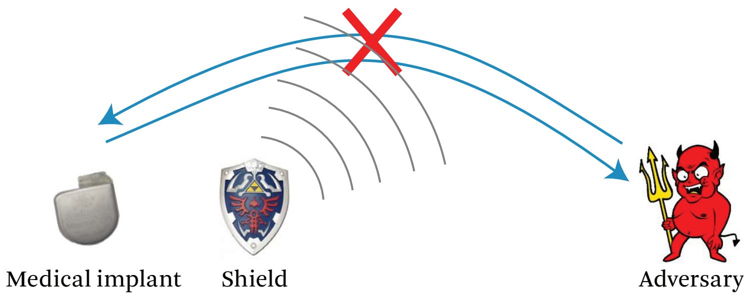


Figure 1.4 **IMDShield.** The external device (shield) transmits a random signal that interferes with the implant's transmission, preventing the adversary from decoding it. Further, the external device leverages our full-duplex radio design to decode the implant's transmission despite interference.

jam and receive simultaneously. Thus, we design a novel duplex radio that can act as a jammer-cum-receiver. This design allows the shield to jam the IMD's messages, while being able to decode them.

Relation to Prior Work. IMDShield is the first system that secures implanted medical devices without modifying them. Prior work on the topic [Halperin et al. 2008b, Schechter 2010] assumes that secure communication requires the transmitter to encrypt transmissions and the receiver to decrypt these transmissions. In contrast, IMDShield leverages interference to design a communication system where the receiver (shield) encrypts the transmissions on behalf of the transmitter (implant). IMDShield also builds on, and contributes to, the emerging topic of full-duplex radios [Duarte and Sabharwal 2010, Choi et al. 2010]. In comparison to prior work in this area, ours differs in that it is the first to demonstrate the value of using full-duplex radios for security and it presents a compact portable design, significantly smaller than its predecessor [Choi et al. 2010].

1.1.4 Secure Pairing Without Passwords or Prior Secrets

In the previous section, we leverage physical-layer signal transmissions to secure implanted wireless device. But what about more typical wireless devices? Establishing secure connections between wireless devices is a general problem that we face while pairing our laptop with a wireless router, a wireless headset with a cellphone, or a home surveillance sensor with its reader. The goal is to ensure that an adversary cannot get access to any of these devices. The traditional approach to achieve this goal requires the user to enter or validate some form of a shared secret such as a password. However, this is difficult for two main reasons. First, ordinary users struggle with picking long and random passwords and often choose vulnerable passwords [Kelton Research 2006, WiFi Alliance 2006, Norman 2009]. Second, even if the user can pick the right kind of a password, there are many devices, e.g., headsets, home surveillance sensors, and medical sensors, which do not have the interface to enter these passwords. So the question we ask is this: Can we establish secure wireless pairing with no pre-shared keys or passwords?

The key challenge in designing such a system is the well-known *man-in-the-middle attack*. Suppose we have Alice and Bob who want to pair with each other, and suppose Alice transmits a pairing message to Bob. Since wireless is a broadcast medium, an adversary can easily tamper with this message to impersonate Alice to Bob and pair with Bob, and vice versa.

Past work assumes that attackers can arbitrarily tamper with wireless messages and as a result they do not trust any messages on the wireless channel. They instead require passwords or other out-of-band channels [Capkun et al. 2008, McCune et al. 2005, Stajano and Anderson 1999, Goodrich et al. 2006, Roth et al. 2008, Mayrhofer and Gellersen 2007]. We question this basic assumption.

In particular, we observe that an attacker cannot eliminate the energy that a random signal produces on the wireless medium. Hence, the attacker cannot hide the fact that such a signal was transmitted. Leveraging this simple idea, we construct a new secure message type that can neither be altered nor hidden without detection. We call such a message a *tamper-evident message*. Now that we can detect tampering we can trust that certain messages have not suffered tampering, and hence use these trusted messages to establish secure wireless connections.

Figure 1.5 shows a simplified version of our tamper-evident message that cannot be altered (but can be hidden). To ensure that an adversary cannot alter the payload of Alice's message, we force any tamper-evident message to include silence periods.

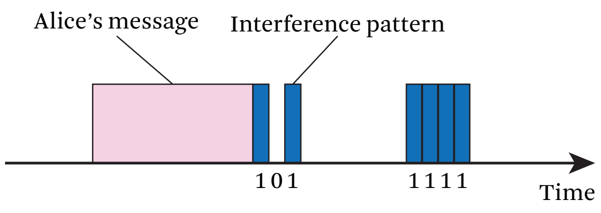


Figure 1.5 **The format of a tamper-evident message.**

As shown in Figure 1.5, the payload of the message is followed by a sequence of short equal-size slots. The transmission of a random signal in a slot is interpreted as a "1" bit, and an idle medium is interpreted as a "0" bit. The bit sequence produced by the slots must match a hash of the message's payload. If an adversary overwrites Alice's message with his own, he must transmit slots corresponding to a hash of his message, including staying silent during any zero hash bits. However, since the hash of the attacker's message differs from that of Alice's message, Alice's message will show up on the medium during the attacker's "0" slots. The attacker cannot eliminate the energy produced by Alice's random signal during her "1" slots. Hence, Bob will detect a mismatch between the slots and the message hash and reject the tampered message.

By exploiting a deeper understanding of how signals interact as they interfere on the wireless medium, we can design a wireless message that cannot be altered without being detected at the receiver. Of course, instead of altering Alice's message, an adversary can simply jam and hide Alice's message. In Chapter 5, we build on the above idea, and design *tamper-evident messages*, which can neither be altered nor hidden, without being detected at the receiver. We then design TEP, a protocol to establish wireless connections that is provably secure against man-in-the-middle attacks. Finally, we integrate TEP with the existing WiFi stack and show how to implement TEP with off-the-shelf WiFi radios.

Relation to Prior Work. TEP is the first wireless pairing protocol that works in-band, with no pre-shared keys, and protects against man-in-the-middle attacks. Over the past decade, there has been significant work on secure pairing without passwords. Past efforts to address this problem, however, rely on prior key distribution or out-of-band communication channels [McCune et al. 2005, Stajano and Anderson 1999, Goodrich et al. 2006, Roth et al. 2008, Mayrhofer and Gellersen 2007, Capkun et al. 2008]. For example, devices can exchange keys over a visual channel between an LCD and a camera [McCune et al. 2005], an audio channel [Goodrich et al. 2006], an infrared channel [Balfanz et al. 2004], a dedicated wireless channel allocated exclusively for key exchange [Capkun et al. 2008], etc. In contrast, by understanding how signals interact when they interfere, we establish secure connections without any passwords, prior key distribution, or out-of-band communication channels.

1.2 Organization

The rest of this book is organized as follows. In Chapter 2, we describe ZigZag decoding in more detail and how it deals with realistic wireless channels and radio imperfections. Chapter 3 describes how TIMO decodes in the presence of cross-technology interference. We first present a measurement study of the impact of cross-technology interference in today's WiFi networks. We then describe TIMO's algorithms and prototype implementation. Next, Chapter 4 presents IMDShield and describes in detail our full-duplex radio design. Further, it demonstrates how the system both protects the confidentiality of the implant's transmitted data and provides access control, preventing an adversary from sending unauthorized commands to the implant. Finally, Chapter 5 describes our tamper-evident message primitive in more detail and builds on this primitive to develop a protocol for securing wireless connections against man-in-the-middle attacks.

2

Decoding 802.11 Collisions

Collisions are a known problem in 802.11 networks [Cheng et al. 2006, Khurana et al. 1998, Judd and Steenkiste 2005, Ng et al. 2005, Ware et al. 2000]. Current 802.11 networks rely on carrier sense (CSMA) to limit collisions; i.e., senders sense the medium and abstain from transmission when the medium is busy. This approach is successful in many scenarios, but when it fails, the impact on the interfering senders is drastic. To see this, consider the hidden terminal scenario in Figure 2.1. Here Alice and Bob are not in range, and hence cannot sense each other. Thus, Alice and Bob are hidden with respect to each other. In such scenarios, the senders either repeatedly collide and their throughputs plummet, or one sender captures the medium, preventing the other from getting packets through [Khurana et al. 1998, Judd and Steenkiste 2005, Ware et al. 2000]. The 802.11 standard proposes the use of RTS-CTS to counter collisions, but experimental results show that enabling RTS-CTS significantly reduces the overall throughput [Judd and Steenkiste 2005, Ware et al. 2000, Xu et al. 2003, Ng et al. 2005], and hence access point (AP) manufacturers disable RTS-CTS by default [Linksys 2005, Netgear 2005]. Ideally, one would like to address this problem without changing the 802.11 medium access protocols (MAC) or affecting senders that do not suffer from hidden terminals.

In this chapter, we introduce ZigZag, a new 802.11 receiver that increases wireless network's resilience to collisions. ZigZag requires no changes to the 802.11 MAC and introduces no overheard in the case of no collision. In fact, in the absence of collisions, ZigZag acts like a typical 802.11 receiver. But, when senders collide, ZigZag achieves the same performance as if the colliding packets were a priori scheduled in separate time slots.

ZigZag exploits a subtle opportunity for resolving collisions, an opportunity that arises from two basic characteristics of 802.11.

1. An 802.11 sender retransmits a packet until it is acked or timed out, and hence when two senders collide they tend to collide again on the same packets.

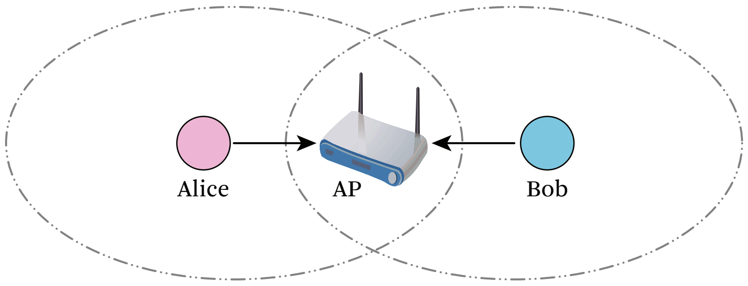


Figure 2.1 **A Hidden Terminals Scenario.**

2. 802.11 senders jitter every transmission by a short random interval,[1] and hence collisions start with a random stretch of interference-free bits.

To see how ZigZag works, consider again the scenario in Figure 2.1, where Alice and Bob, unable to sense each other, transmit simultaneously, causing collisions. When Alice's packet collides with Bob's, both senders retransmit their packets causing a second collision, as shown in Figure 2.2. Further, because of 802.11 random jitters, the two collisions are likely to have different offsets, i.e., $\Delta_1 \neq \Delta_2$. Suppose that the AP can compute these offsets (as explained in Section 2.4.1), the AP can then find a chunk of bits that experience interference in one collision but is interference-free in the other, such as chunk 1 in Figure 2.2. A ZigZag AP uses this chunk to bootstrap its decoder. In particular, since chunk 1 is interference-free in the first collision, the AP can decode it using a standard decoder. The AP then subtracts chunk 1 from the second collision to decode chunk 2. Now, it can go back to the first collision, subtract chunk 2, decode chunk 3, and proceed until both packets are fully decoded.

ZigZag's key contribution is a novel approach to resolving interference, different from prior work on interference cancellation [Verdu 1998, Hou et al. 2006] and joint decoding [Tse and Vishwanath 2005]. Basic results on the capacity of the multi-user channel show that if the two hidden terminals transmit at the rate supported by the medium in the absence of interference, i.e., rate R in Figure 2.3, the aggregate information rate in a collision, being as high as $2R$, exceeds capacity, precluding any decoding [Tse and Vishwanath 2005, Gallager 1985]. Thus, state-of-the-art interference cancellation and joint decoding, designed for cellular networks with non-bursty traffic and known users [Verdu 1998, Andrews 2005], have a fundamental limitation when applied in 802.11 networks: they require a sender to change the way it modulates and

1. Each transmission picks a random slot between 0 and CW [IEEE 2012].

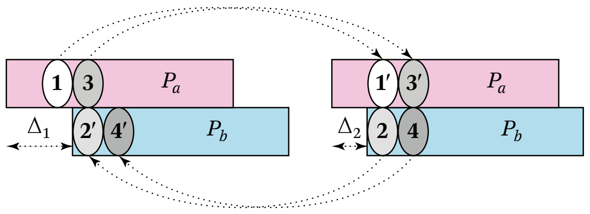


Figure 2.2 **ZigZag decoding.** ZigZag decodes first chunk 1 in the first collision, which is interference-free. It subtracts chunk 1 from the second collision to decode chunk 2, which it then subtract from the first collision to decode chunk 3, etc.

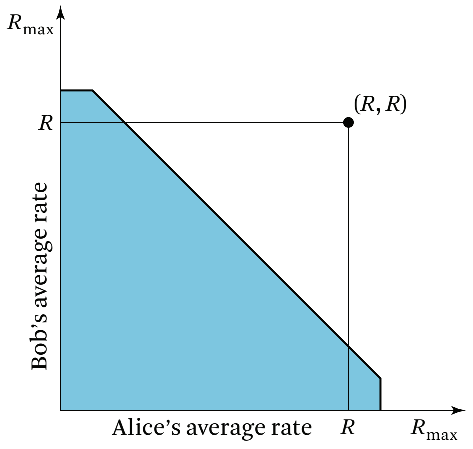


Figure 2.3 **Standard interference cancellation and joint decoding require inefficient rates.** The figure shows the capacity region of the multi-user channel. If Alice and Bob transmit close to the best rate supported by the medium in the absence of interference, R, their combined rates will be (R, R), which is outside the capacity region, and hence cannot be decoded.

codes a packet according to whether the packet will collide or not. This leaves 802.11 senders with the following tradeoff: either they tune to a suboptimal rate that works in the presence of collision, although not every packet will collide, or they send at the best rate in the absence of collision, but accept that the network cannot use these methods to resolve collisions. In contrast, with ZigZag, the senders need not make such a trade-off. ZigZag allows the senders to transmit at the best rate supported by the medium in the absence of collisions. However, if collisions occur, ZigZag decodes pairs of collisions that contain the same packets. The average information rate in such a collision pair is $2R/2 = R$. This rate is both decodable and as efficient as if the two packets were scheduled in separate time slots.

ZigZag has the following key features.

It is modulation independent. In ZigZag, every chunk is first rid of interference then decoded. Hence, ZigZag can employ a standard 802.11 decoder as a black-box, which allows it to work with collisions independent of their underlying modulation scheme (i.e., bit rate), and even when the colliding packets are modulated differently.

It is backward compatible. A ZigZag receiver can operate with unmodified 802.11 senders and requires no changes to the 802.11 protocol (see Section 2.6).

It generalizes to more than a pair of colliding packets. Explained in Section 2.7 and experimentally demonstrated in Section 2.9.6.

We have implemented a ZigZag prototype in GNU Radio, and evaluated it in a 14-node testbed, where 10% of the sender–receiver pairs are hidden terminals, 10% sense each other partially, and 80% sense each other perfectly. Our results reveal the following findings.

- The loss rate averaged over scenarios with partial or perfect hidden terminals decreases from 72.6% to less than 0.7%, with some severe cases where the loss rate goes down from 100% to 0.
- Averaging over all sender–receiver pairs, including those that do not suffer from hidden terminals, we find that ZigZag improves the average throughput by 25.2% when compared to current 802.11.
- Our BPSK implementation and our 4-QAM and 16-QAM simulations show that ZigZag and collision-free decoding achieve the same bit error (BER) for comparable SNRs. Surprisingly, at BPSK and 4-QAM, ZigZag has a slightly lower BER than if the two packets were collision-free. This is because, in ZigZag, every bit is received twice, once in every collision, improving its chances of being correctly decoded.

2.1 Related Work

Related work falls into the following two areas.

(a) Collisions in WLAN and Mesh Networks. Recent work [Halperin et al. 2007, Halperin et al. 2008a] advocates the use of successive interference cancellation (SIC) and joint decoding to resolve 802.11 collisions. As explained later these schemes work only when the colliding senders transmit at a bit rate (i.e., information rate) significantly lower than allowed by their respective SNRs and code redundancy. The authors have built a ZigBee prototype of successive interference cancellation [Halperin et al. 2008a]. Since ZigBee has no rate adaptation and employs a high redundancy code (every 4 bits are expanded to 32 bits), it experiences scenarios in which the bit rate is significantly below what can be supported by the SNR and the code rate. In such scenarios, SIC could significantly improve the throughput. In contrast, ZigZag works even when a sender uses a bit rate that matches its channel's SNR and the redundancy of its code (as would be the case for systems with proper rate adaptation). In that respect, ZigZag provides an attractive alternative to SIC.

Our work is also related to our prior work on analog network coding (ANC) [Katti et al. 2007]. An ANC receiver, however, can decode collisions only if it already knows one of the two colliding packets. It cannot deal with general collisions or hidden terminals. In principle, one can combine ANC and ZigZag to create a system both addresses hidden terminals, and collects network coding gains.

Additionally, prior works have studied wireless interference [Reis et al. 2006, Gummadi et al. 2007, Cheng et al. 2006, Khurana et al. 1998, Judd and Steenkiste 2005, Ng et al. 2005, Ware et al. 2000], and proposed MAC modifications to increase resilience to collisions [Zhu et al. 2007, Fullmer and Garcia-Luna-Aceves 1997, Karn 1990, Bharghavan et al. 1994, Muqattash and Krunz 2003]. In comparison, ZigZag presents mechanisms that decode collisions rather than avoiding them, and works within the 802.11 MAC rather than proposing a new MAC.

(b) Communication and Information Theory. The idea of decoding interfering users has received much interest in information and communications theories [Tse and Vishwanath 2005, Verdu 1998, Castoldi 2002, Tse et al. 2004, Viterbi 1990]. The main feature that distinguishes ZigZag from prior works in those areas is that ZigZag resolves 802.11 collisions without requiring any scheduling, power control, synchronization assumptions, or coding.

Among the deployed systems, CDMA receivers decode a user by treating all other users as noise [Castoldi 2002]. A CDMA solution for hidden terminals in WLANs, however, would require major changes to 802.11 including the use of power control and special codes [Andrews 2005, Castoldi 2002]. Furthermore, CDMA is known to be highly suboptimal in high SNR regimes (e.g., worse than TDMA [Tse and Vishwanath 2005]), which are typical in WLANs.

Finally, successive interference cancellation (SIC) has been used to decode interfering users in CDMA cellular networks [Andrews 2005]. SIC requires the interfering senders to have significantly different powers [Verdu 1998] or different levels of coding [Hou et al. 2006, Tse and Vishwanath 2005]. It also requires tight control from the base station to ensure that the total information rate stays below capacity. Conceptually, SIC may be perceived as a special case of ZigZag, in which a chunk is a full packet, i.e., a full packet is decoded and subtracted from the collision signal to decode the other packet. However, by iterating over strategically picked chunks, ZigZag can resolve interference even when the colliding senders have similar SNRs, are not coordinated, and do not use special codes.

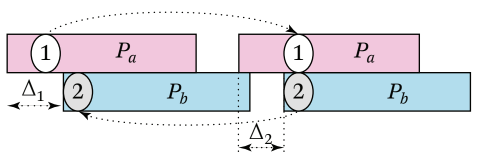
2.2 Scope

ZigZag is an 802.11 receiver design that decodes collisions. It focuses on hidden terminals in WLANs. ZigZag's benefits extend to mesh networks, where having receivers that can decode collisions could enable more concurrent transmissions and hence higher spatial reuse. Exploring mesh benefits is, however, beyond the scope of this book.

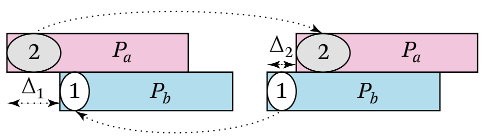
ZigZag adopts a best effort design; in the absence of collisions it acts like current 802.11 receivers, but when collisions occur it tries to decode them. Of course there are collision patterns that ZigZag cannot decode and there are cases where, although the pattern is decodable, decoding may fail because of insufficient SNR. However, since ZigZag does not introduce any overhead for the case of no collision, its presence can only increase the throughput of the WLAN. In Section 2.6, we explain how one can deploy ZigZag in a WLAN by changing only the access points and without modifying the clients.

ZigZag resolves a variety of collision patterns. The main idea underlying its decoding algorithm is to find a collision-free chunk, which it exploits to bootstrap the decoding process. Once the decoder is bootstrapped the process is iterative and at each stage it produces a new interference-free chunk, decodable using standard decoders. For example, as explained in Section 2, ZigZag can decode the pattern in Figure 2.2 by decoding first chunk 1 in the first collision, and subtracting it from the second collision, obtaining chunk 2, which it decodes and subtracts from the first collision, etc. Using the same principle, ZigZag can decode other patterns like those in Figure 2.4. In particular, it can decode patterns where the collisions overlap as in Figure 2.4a, and patterns in which colliding packets change order as in Figure 2.4b, or even patterns where the packets have different sizes, as in Figure 2.4c.

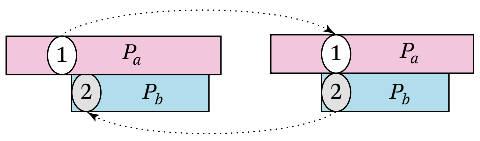
ZigZag also exploits collision patterns that arise from capture effects. Suppose that Alice's power at the AP is significantly higher than Bob's, and hence her packets enjoy the capture effect [Ware et al. 2000]. Currently, such a scenario translates into significant unfairness to Bob, whose packets do not get through [Khurana et al. 1998, Judd and Steenkiste 2005, Ware et al. 2000]. Like current APs, a ZigZag AP decodes every packet from Alice, the high-power sender. Unlike current APs, however, ZigZag



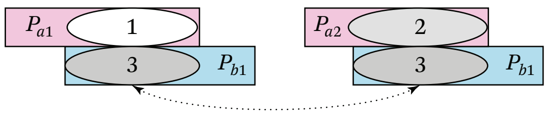
(a) Overlapped collisions



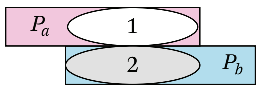
(b) Flipped order



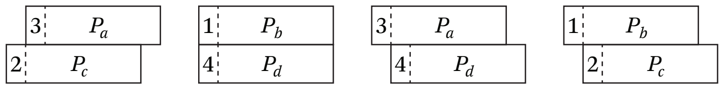
(c) Different packet sizes



(d) Alice's packets enjoy the capture effect



(e) Single decodable collision; inefficient choice of bit rates



(f) Nodes *A* and *B* are hidden from *C* and *D*

Figure 2.4 **ZigZag applies to various collision patterns.** Subscripts refer to a packet's sender and id, e.g., P_{a1} is Alice's first packet. The top three patterns are decoded chunk by chunk. The forth pattern may occur when Alice's SNR is significantly higher than Bob's. The fifth pattern occurs when Alice's SNR is higher than Bob's, and the bit rates are too low for the SNRs. The last pattern occurs when two groups of nodes are hidden from each other.

subtracts Alice's packet from the collision signal and try to decode Bob's packet. However, if Alice's power is excessively high, even a small imperfection in subtracting her signal would contribute a significant noise to Bob's, preventing correct decoding of his packets. In this case, the next collision will involve a new packet from Alice and Bob's retransmission of the same packet, as shown in Figure 2.4d. ZigZag decodes Alice's new packet and subtracts it to obtain a second version of Bob's packet, which may also contain errors. ZigZag, however, combines the two faulty versions of Bob's packet to correct the errors. This is done using Maximal Ratio Combining (MRC) [Brennan 1955], a classic method for combining information from two receptions to correct for bit errors.

In addition, whenever the powers permit, ZigZag decodes patterns that involve a single collision like those in Figure 2.4e. This occurs when Alice's power is significantly higher than Bob's, and both senders happen to transmit at a bit rate lower than the best rate supported by the channel. In this case, ZigZag can apply standard successive interference cancellation [Verdu 1998], i.e., ZigZag decodes P_a and subtracts it from the received signal to decode P_b, decoding both packets using a single collision. As explained in Section 2.1, successive interference cancellation is a special case of ZigZag, in which a chunk is a full packet. This special case applies only when the bit-rate is too low given the senders' SNRs, and one of the senders has significantly more SNR than the other.

ZigZag can also decode patterns that involve more than two nodes, like that in Figure 2.4f. This pattern may occur when two groups of nodes cannot sense each other. For example, nodes A and B, which are in the same room, can sense each other, but cannot sense nodes C and D, which happen to be in a different room.

ZigZag can also decode collisions that involve more than a pair of packets, which is discussed in Section 2.7.

2.3 A Communication Primer

A wireless signal is typically represented as a stream of discrete complex numbers [Meyr et al. 1998]. To transmit a packet over the wireless channel, the transmitter maps the bits into complex symbols, in a process called *modulation*. For example, the BPSK modulation (used in 802.11 at low rates) maps a "0" bit to $e^{j\pi} = -1$ and a "1" bit to $e^{j0} = 1$. The transmitter generates a complex symbol every T seconds. In this chapter, we use the term $\mathbf{x}[n]$ to denote the complex number that represents the nth transmitted symbol.

The received signal is also represented as a stream of complex symbols spaced by the sampling interval T. These symbols differ, however, from the transmitted symbols, both in amplitude and phase. In particular, if the transmitted symbol is $\mathbf{x}[n]$ the received symbol can be approximated as

$$\mathbf{y}[n] = \mathbf{H}\mathbf{x}[n] + \mathbf{w}[n], \tag{2.1}$$

where $\mathbf{H} = he^{\gamma}$ is also a complex number, whose magnitude h refers to channel attenuation and its angle γ is a phase shift that depends on the distance between the transmitter and the receiver, and $\mathbf{w}[n]$ is a random complex noise.[2]

If Alice and Bob transmit concurrently their signals add up, and the received signal can be expressed as

$$\mathbf{y}[n] = \mathbf{y}_A[n] + \mathbf{y}_B[n] + \mathbf{w}[n],$$

where $\mathbf{y}_A[n] = \mathbf{H}_A\mathbf{x}_A[n]$ and $\mathbf{y}_B[n] = \mathbf{H}_B\mathbf{x}_B[n]$ refer to Alice's and Bob's signals after traversing their corresponding channels to the AP. Note that the above does not mean that we assume the nth symbol from Alice combines with the nth symbol from Bob. The notation is only to keep the exposition clear.

2.3.1 Practical Issues

A few practical issues complicates the process of estimating the transmitted symbols from the received symbols: frequency offset, sampling offset, and inter-symbol interference. Typically, a decoder has built-in mechanisms to deal with these issues [Meyr et al. 1998].

(a) Frequency Offset and Phase Tracking. It is virtually impossible to manufacture two radios centered at the same exact frequency. Hence, there is always a small frequency difference, δf, between transmitter and receiver. The frequency offset causes a linear displacement in the phase of the received signal that increases over time, i.e.,

$$\mathbf{y}[n] = \mathbf{H}\mathbf{x}[n]e^{j2\pi n\delta f T} + \mathbf{w}[n].$$

Typically, the receiver estimates δf and compensates for it.

(b) Sampling Offset. The transmitted signal is a sequence of complex samples separated by a period T. However, when transmitted on the wireless medium, these discrete values have to be interpolated into a continuous signal. The continuous signal is equal to the original discrete samples only if sampled at the exact same positions where the discrete values were. Due to lack of synchronization, a receiver cannot sample the received signal exactly at the right positions. There is always a sampling offset, μ. Further,

2. This models flat-fading quasi-static channels.

the drift in the transmitter's and receiver's clocks results in a drift in the sampling offset. Hence, decoders have algorithms to estimate μ and track it over the duration of a packet.

(c) Inter-Symbol Interference (ISI). While Equation 2.1 makes it look as if a received symbol $\mathbf{y}[n]$ depends only on the corresponding transmitted symbol $\mathbf{x}[n]$, in practice, neighboring symbols affect each other to some extent. Practical receivers apply linear equalizers [Lee and Messerschmitt 1988] to mitigate the effect of ISI.

2.4 ZigZag Decoding

We explain ZigZag decoding using the hidden terminal scenario in Figure 2.6, where Alice and Bob, not able to sense each other, transmit simultaneously to the AP, creating repeated collisions. Later in Section 2.7, we extend our approach to a larger number of colliding senders.

Like current 802.11, when a ZigZag receiver detects a packet it tries to decode it, assuming no collision, and using a typical decoder. If decoding fails (e.g., the decoded packet does not satisfy the checksum), the ZigZag receiver will check whether the packet has suffered a collision, and proceed to apply ZigZag decoding.

2.4.1 Is It a Collision?

To detect a collision, the AP exploits that every 802.11 packet starts with a known preamble [IEEE 2012]. The AP detects a collision by correlating the known preamble with the received signal. Correlation is a popular technique in wireless receivers for detecting known signal patterns [Castoldi 2002]. Suppose that the known preamble is L samples. The AP aligns these L samples with the first L received samples, computes the correlation, shifts the alignment by one sample, and recomputes the correlation. The AP repeats this process until the end of the packet. The preamble is a pseudorandom sequence that is independent of shifted versions of itself, as well as Alice's and Bob's data. Hence, the correlation is near zero except when the preamble is perfectly aligned with the beginning of a packet. Figure 2.5 shows the correlation as a function of the position in the received signal. The measurements are collected using GNURadios (see Section 2.9). Note that when the correlation spikes in the middle of a reception, it indicates a collision. Further, the position of the spike corresponds to the beginning of the second packet, and hence shows Δ, the offset between the packets.

The above argument is only partially correct because the frequency offset can destroy the correlation, unless the AP compensates for it. Assume that Alice's packet starts first and Bob's packet collides with it starting at position Δ. To detect Bob's colliding

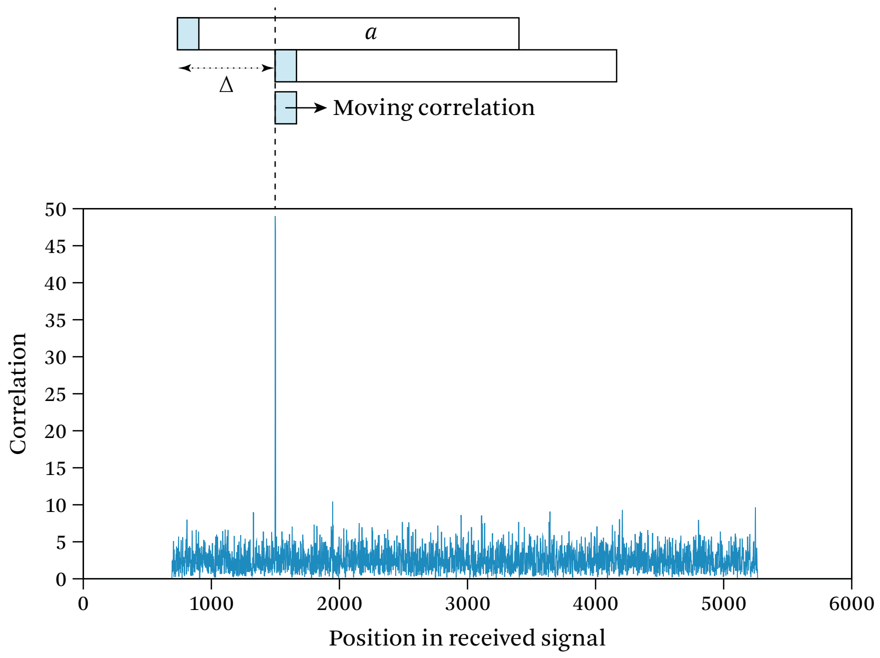


Figure 2.5 **Detecting collisions by correlation with the known preamble.** The correlation spikes when the correlated preamble sequence aligns with the preamble in Bob's packet, allowing the AP to detect the occurrence of a collision and where it starts.

packet, the AP has to compensate for the frequency offset between Bob and itself. The frequency offset does not change over long periods, and thus the AP can maintain coarse estimates of the frequency offsets of active clients as obtained at the time of association. The AP uses these estimates in the computation.

Mathematically, the correlation is computed as follows. Let $\mathbf{y}$ be the received signal, which is the sum of the signal from Alice, $\mathbf{y}_A$, the signal from Bob, $\mathbf{y}_B$, and the noise term $\mathbf{w}$. Let the samples $\mathbf{s}[k], 1 \leq k \leq L$, refer to the known preamble, and $\mathbf{s}^*[k]$ be the complex conjugate. The correlation, Γ, at position Δ is

$$\begin{aligned}\Gamma(\Delta) &= \sum_{k=1}^{L} \mathbf{s}^*[k]\mathbf{y}[k+\Delta] \\ &= \sum_{k=1}^{L} \mathbf{s}^*[k](\mathbf{y}_A[k+\Delta] + \mathbf{y}_B[k] + \mathbf{w}[k]).\end{aligned}$$

The preamble, however, is independent of Alice's data and the noise, and thus the correlation between the preamble and these terms is about zero. Since Bob's first L samples are the same as the preamble, we obtain

$$\begin{aligned}\Gamma(\Delta) &= \sum_{k=1}^{L} \mathbf{s}^*[k]\mathbf{y}_B[k] \\ &= \sum_{k=1}^{L} \mathbf{s}^*[k]\mathbf{H}_B\mathbf{s}[k]e^{j2\pi k\delta f_B T} \\ &= \mathbf{H}_B \sum_{k=1}^{L} |\mathbf{s}[k]|^2 e^{j2\pi k\delta f_B T}.\end{aligned}$$

Since a frequency offset exists between Bob and the AP, i.e., $\delta f_B \neq 0$, the terms inside the sum have different angles and may cancel each other. Thus, the AP should compute the value of the correlation after compensating for the frequency offset, which we call Γ'. At position Δ this value becomes

$$\begin{aligned}\Gamma'(\Delta) &= \mathbf{H}_B \sum_{k=1}^{L} |\mathbf{s}[k]|^2 e^{j2\pi k\delta f_B T} \times e^{-j2\pi k\delta f_B T} \\ &= \mathbf{H}_B \sum_{k=1}^{L} |\mathbf{s}[k]|^2.\end{aligned}$$

The magnitude of $\Gamma'(\Delta)$ is the sum of energy in the preamble, and thus it is significantly large, i.e., after compensating for the frequency offset, the magnitude of the correlation spikes when the preamble aligns with the beginning of Bob's packet, as shown in Figure 2.5. Imposing a threshold enables us to detect whether the AP received a collision signal and where exactly the second packet starts.

2.4.2 Did the AP Receive Two Matching Collisions?

Now that it is clear that the received signal is the result of collision, the AP searches for a matching collision, i.e., a collision of the same two packets. The AP stores recent unmatched collisions (i.e., stores the received complex samples). It is sufficient to store the few most recent collisions because, in 802.11, colliding sources try to retransmit a failed transmission as soon as the medium is available [IEEE 2012].

We use the same correlation trick to match the current collision against prior collisions. Assume the AP is trying to match two collisions (P_1, P_2), and (P_1', P_2'). Without loss of generalization, let us focus on checking whether P_2 is the same as P_2'. The AP already knows the offset in each collision, i.e., Δ and Δ'. The AP aligns the two collisions

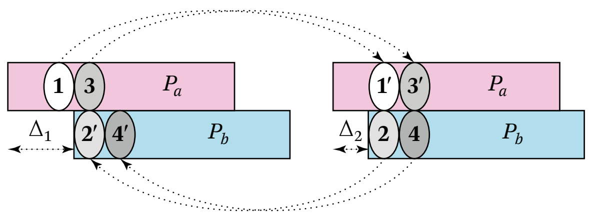


Figure 2.6 **ZigZag decodes then re-encodes a chunk.** Before subtracting a decoded chunk, like chunk 1, ZigZag needs to re-encode the bits to create an image of chunk 1′, as received in the second collision.

at the positions where P_2 and P_2' start. If the two packets are the same, the samples aligned in such a way are highly dependent (they are the same except for noise and the retransmission flag in the 802.11 header), and thus the correlation spikes. If P_2 and P_2' are different, their data is not correlated and the correlation does not spike at that alignment.

2.4.3 How Does the AP Decode Matching Collisions?

Say that the AP found a pair of matching collisions like those in Figure 2.6. Note that Figure 2.6 is the same as Figure 2.2 in the introduction except that we distinguish between two images of the same chunk that occur in different collisions, e.g., chunk 1 and chunk 1′. By now the AP knows the offsets Δ_1 and Δ_2, and hence it can identify all interference-free symbols and decode them using a standard decoder.

Next, the AP performs ZigZag decoding, which requires identifying a *bootstrapping chunk*, i.e., a sequence of symbols marred by interference in one collision and interference-free in the other. Suppose that the first collision has the larger offset, i.e., $\Delta_1 > \Delta_2$; the bootstrapping chunk then is located in the first collision starting at position Δ_2 and has a length of $\Delta_1 - \Delta_2$ samples. This is chunk 1 in Figure 2.6.

The rest of the decoding works iteratively. In each iteration, the AP decodes a chunk, re-encodes the decoded symbols and subtract them from the other collision. For example, in Figure 2.6, the AP decodes chunk 1 from the first collision, re-encodes the symbols in chunk 1 to create an image of chunk 1′, which it subtracts from the second collision to obtain chunk 2. The AP iterates on the rest of the chunks as it did on chunk 1, until it is done decoding all chunks in the colliding packets.

(a) The Decoder. ZigZag can use any standard decoder as a black box. Specifically, the decoder operates on a chunk after it has been rid from interference, and hence can use standard techniques. This characteristic allows ZigZag to directly apply to any modulation scheme as it can use any standard decoder for that modulation as a black box. Further, the two colliding packets may use different modulation (different bit rates) without requiring any special treatment.

(b) Re-Encoding a Chunk. Now that the AP knows the symbols that Alice sent in chunk 1, it uses this knowledge to create an estimate of how these symbols would look after traversing Alice's channel to the AP, i.e., to create an image of chunk $1'$, which it can subtract from the second collision.

In Section 2.4.4 we explain how the AP computes channel parameters, but for now, let us assume that the AP knows Alice's channel, i.e., $\mathbf{H}_A$, δf_A, and μ_A. Denote the symbols in chunk 1 by $\mathbf{x}_A[n] \dots \mathbf{x}_A[n+K]$. A symbol that Alice sends, $\mathbf{x}_A[n]$, is transformed by the channel to $\mathbf{y}_A[n]$, where

$$\mathbf{y}_A[n] = \mathbf{H}_A \mathbf{x}_A[n] e^{j2\pi\delta f_A T}. \tag{2.2}$$

The AP would have received $\mathbf{y}_A[n]$ had it sampled the signal exactly at the same locations as Alice. Because of sampling offset, the AP samples the received signal μ_A seconds away from Alice's samples. Thus, given the samples $\mathbf{y}_A[n] \dots \mathbf{y}_A[n+K]$, the AP has to interpolate to find the samples at $\mathbf{y}_A[n+\mu_A] \dots \mathbf{y}_A[n+K+\mu_A]$.

To do so, we leverage that we have a band-limited signal sampled according to the Nyquist criterion. Nyquist says that for such signals, one can interpolate the signal at any discrete position, e.g., $n+\mu_A$, with complete accuracy, using the following equation [Meyr et al. 1998]:

$$\mathbf{y}_A[n+\mu_A] = \sum_{i=-\infty}^{\infty} \mathbf{y}_A[i] \operatorname{sinc}(\pi(n+\mu_A-i)),$$

where sinc is the sinc function. In practice, the above equation is approximated by taking the summation over few symbols (about eight symbols) in the neighborhood of n.

Now that the AP has an image of chunk $1'$ as received, it subtracts it from the second collision to obtain chunk 2, and proceeds to repeat the same process on this latter chunk.

2.4.4 Estimating and Tracking System Parameters

The receiver estimates the system's parameters using the preamble in Alice's and Bob's packets. Without loss of generality, we focus on Bob, i.e., we focus on the sender that starts second. This is the harder case since the preamble in Bob's packet, typically used for channel estimation, is immersed in noise. We need to learn $\mathbf{H}_B$, μ_B, and δf_B.

(a) Channel. Again we play our correlation trick, i.e., we correlate the received samples with the known preamble. Recall that the correlation at the peak is

$$\Gamma'(\Delta) = \mathbf{H}_B \sum_{k=1}^{L} |\mathbf{s}[k]|^2.$$

The AP knows the magnitude of the transmitted preamble i.e., it knows $|\mathbf{s}[k]|^2$. Hence, once it finds the maximum value of the correlation over the collision, it substitutes in the above equation to compute $\mathbf{H}_B$.

(b) Frequency Offset. The frequency offset does not change significantly. Since decoders already estimate the frequency offset, an initial coarse estimate can be computed using any prior interference-free packet from the client (e.g., the association packet).

This coarse estimate, however, is not sufficient since any residual errors in estimating δf translate into linear displacement in the phase that accumulates over the duration of a packet. Any typical decoder tracks the signal phase and corrects for the residual errors in the frequency offset. Since ZigZag uses a typical decoder as a black box, it need not worry about tracking the phase while decoding. However, as it reconstructs an image of a received chunk, ZigZag tracks the phase. Consider as an example, reconstructing an image of chunk $1'$. First we reconstruct the image using the current estimate of the frequency offset, as explained in Section 2.4.3(b). Next, we subtract that image from the second collisions to get chunk 2. Now, we reconstruct chunk 2 and subtracted from the second collision, creating an estimate of chunk $1'$, which we term chunk $1''$. We compare the phases in chunk $1'$ and chunk $1''$. The difference in the phase is caused by the residual error in our estimate of the frequency offset. We update our estimate of the frequency offset as follows:

$$\delta f = \delta f + \alpha \delta\phi / \delta t,$$

where α is just a small multiplier and $\delta\phi$ is the phase error which accumulated over δt.

(c) Sampling Offset. The procedure used to update and track the sampling offset is fairly similar to that used to update and track the frequency offset. Namely, the black-box decoder tracks the sampling offset when decoding a chunk. When reconstructing the image of a chunk, like chunk $1'$, we use the differences between chunk $1'$ and $1''$ to estimate the residual error in the sampling offset and track it.[3]

3. We use the Muller-and-Muller algorithm [Meyr et al. 1998] to estimate sampling offset errors.

(d) Inter-Symbol Interference. When we reconstruct a chunk to subtract it from the received signal, we need to create as close an image of the received version of that chunk as possible. This includes any distortion that the chunk experienced because of multipath effects, hardware distortion, filters, etc. To do so, we need to invert the linear filter (i.e., the equalizer) that a typical decoder uses to remove these effects. The filter takes as input the decoded symbols before removing ISI, and produces their ISI-free version, as follows:

$$\mathbf{x}[i] = \sum_{l=-L}^{L} h_l\, \mathbf{x}_{ISI}[i+l],$$

where the h_l's are known as the filter taps. For our purpose, we can take the filter from the decoder and invert it. We apply the inverse filter to the symbols $\mathbf{x}[n]$ before using them in Equation 2.2 to ensure that our reconstructed chunk incorporates these distortions.

2.5 Dealing with Errors

Up until now, we have described the system assuming correct decoding. But what happens if the AP makes a mistake in decoding a symbol? For example, in Figure 2.6, suppose the AP mistakenly decodes the first bit in chunk 1 as a "0" bit, when it is actually a "1" bit. Since chunk 1 is subtracted from the second collision to obtain chunk 2, the error will affect the first symbol in chunk 2. This in turn will affect the first symbol in chunk 3, and so on. We will show the following.

- If a symbol error occurs while decoding, it may affect later chunks, but this propagation does not persist. It dies exponentially fast.
- The errors can be further reduced by appling ZigZag in both the forward and backward directions and combining the results.

(a) Errors Die Exponentially Fast. Intuitively, suppose the AP made a random error in decoding a symbol; the error will propagate to subsequent symbols making them random. However, any modulation scheme has only a few possible symbol values (e.g., a BPSK symbol can be either "0" or "1"). Even when a symbol is randomly decoded, there is a reasonable chance the randomly picked value is correct. Thus, a decoding mistake propagates for a stretch of symbols until it is corrected by chance, at which point it stops affecting subsequent symbols. Assuming the probability of randomly picking the right symbol is p, the errors dies at a rate $\frac{1}{p}$.

We formalize the above argument for the case of BPSK, which maps a "0" bit to -1 and a "1" bit to $+1$. Assume the AP makes a mistake in decoding some symbol $\mathbf{y}_A$, and

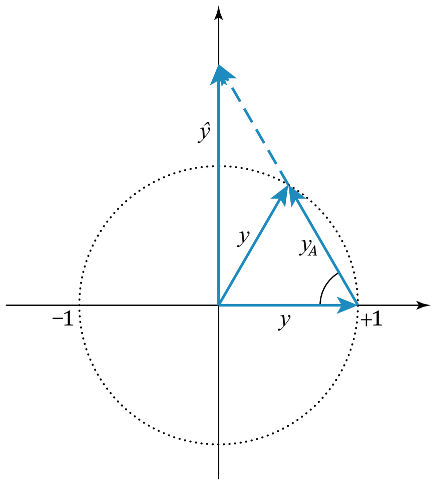


Figure 2.7 **Errors die exponentially fast.** The error causes the AP to sum $\mathbf{y}_A$ instead of subtracting it. Hence, the error propagates from $\mathbf{y}_A$ to the estimate $\hat{\mathbf{y}}_B$, i.e., from one chunk to the next, only when the angle between the vectors is smaller than 60°, which occurs with probability $\frac{1}{3}$.

tries to use the erroneous symbol to decode $\mathbf{y}_B$ by subtracting the decoded vector from the received signal $\mathbf{y} = \mathbf{y}_A + \mathbf{y}_B$.[4] In the worst case, and as shown in Figure 2.7, the error causes the AP to add the vector instead of subtracting it, and hence the AP estimates $\hat{\mathbf{y}}_B$ as $\mathbf{y}_B + 2\mathbf{y}_A$. In BPSK, the AP will decode $\mathbf{y}_B$ to the wrong bit value only if the estimate $\hat{\mathbf{y}}_B$ has the opposite sign of the original vector. This will happen only if the angle between the two vectors $\mathbf{y}_B$ and $\mathbf{y}_A$ is less than $-60°$. The frequency offset between Alice and Bob means that the vectors $\mathbf{y}_B$ and $\mathbf{y}_A$ can have any angle with respect to each other. Thus, the error propagates with probability less than $\frac{60}{180} = \frac{1}{3}$; i.e., in BPSK, errors die exponentially fast at a rate $\frac{2}{3}$.

Figure 2.8 shows a simulation of error propagation in ZigZag. We insert a decoding error by randomly mistaking a symbol as one of its neighbors in the constellation. We compute the number of subsequent symbols that are affected by this error. The figure shows that errors die exponential quickly. The figure, however, shows that errors die faster in BPSK and 4-QAM than in 16-QAM, and hence ZigZag performs better in these modulation schemes.

4. We ignore the noise term $\mathbf{w}$ since it has a random effect on the error and can equally emphasize it or correct it.

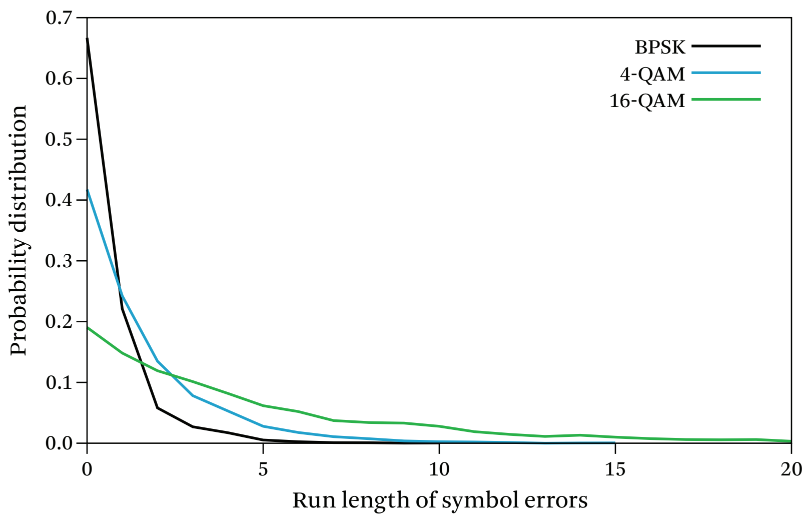


Figure 2.8 **The Probability of Error Propagation Dies Fast.**

(b) Forward and Backward Decoding. The ZigZag algorithm described so far decodes forward. In Figure 2.2, it starts with chunk 1 in the first collision and proceeds until both packets are decoded. However, clearly the figure is symmetric. The AP could wait until it received all samples, then decode backward. If the AP does so, it will have two estimates for each symbol. ZigZag combines these estimates to both combat error propagation and reduce the overall errors. To do so, ZigZag builds on prior results in diversity combining [Woo et al. 2007, Brennan 1955]; whenever there is a mismatch between forward and backward decoding, ZigZag uses the soft values of the decoded symbols as a confidence measure. It picks the results of forward or backward decoding depending on which one has a higher confidence (the details are in [Gollakota and Katabi 2008]). In practice, instead of decoding all the way forward and then backward, one can do it on a chunk-by-chunk basis, using the most recently decoded chunk as a bootstrapping chunk for backward decoding.

2.6 Backward Compatibility

It would be beneficial if ZigZag requires no changes to senders. In this case, one can improve resilience to interference in a WLAN by purely changing the APs, and without requiring any modifications to the clients (e.g., laptops, PCs, PDAs). Compatibility with unmodified 802.11 senders requires a ZigZag receiver to ACK the colliding senders once it decoded their packets; otherwise, the senders will retransmit again unnecessar-

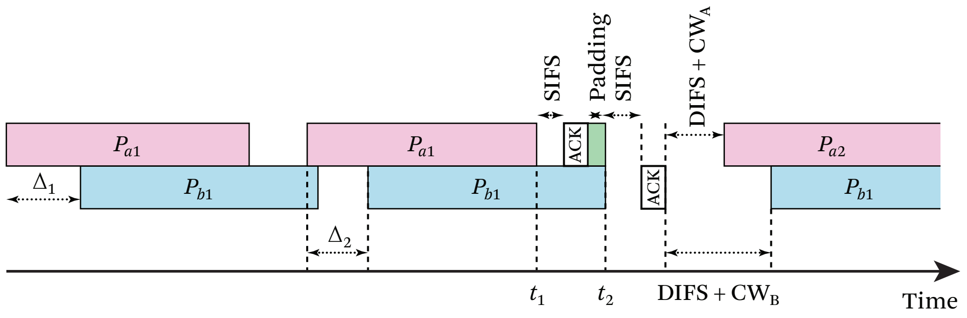


Figure 2.9 **How ZigZag sends 802.11 synchronous ACKs.**

ily. Recall that an 802.11 sender expects the ACK to follow the packet, separated only by a short interval called SIFS [IEEE 2012]: Can a ZigZag receiver satisfy such requirement?

The short answer is "yes, with a high probability." To see how, consider again the example where Alice and Bob are hidden terminals, and suppose that the AP uses ZigZag to decode two of their packets, P_{a1} and P_{b1}, as shown in Figure 2.9. The AP acks the packets according to the scheme outlined in Figure 2.9. Specifically, by time t_1, the AP has fully decoded both P_{a1} and P_{b1}. Even more, by t_1 the AP has performed both forward-decoding and backward decoding for all bits transmitted so far, i.e., all bits except the few bits at the end of P_{b1}.[5] Thus, at t_1 the AP declares both packets decoded. It waits for a SIFS and acks packet P_{a1}. Although the ACK collides with the tail of packet P_{b1}, the ACK will be received correctly because Alice cannot hear Bob's transmission. Bob too will not be disturbed by the AP's ACK to Alice because practical transmitters cannot receive and transmit at the same time. The AP then transmits some random signal to prevent Alice from transmitting her next packet, P_{a2}, before Bob's packet is acked. The AP knows how long this padding signal should be since it already has a decoded version of Bob's packet and knows its length. After Bob finishes his transmission the AP acks him as well.

One question remains, however: Would the offset between the two colliding packets suffice to send an ACK? Said differently, in Figure 2.9, how likely is it that $t_2 - t_1 > SIFS + ACK$. One can show that, given 802.11 timing, the likelihood that the time offset between the two packets is sufficient to send an ACK is quite high. In particular, for the common deployment of backward compatible 802.11g, we prove the following.

5. This assumes the receiver tries in parallel to use standard decoding and ZigZag, and takes whichever satisfies the checksum.

Lemma 2.1 In 802.11g, the probability that the time offset between two colliding packets is sufficient for sending an ACK is higher than 93.7%.

Proof Let us denote the duration of the slot time by S, ACK duration by ACK, SIFS duration by $SIFS$, and the initial congestion window by CW. We need the offset between the two colliding packets in the second collision to be greater than $SIFS + ACK$. Since in the second collision, Alice and Bob randomly pick a slot in the congestion window of size $2CW$, the probability that Alice picks a slot close enough to Bob to have an offset of less than $SIFS + ACK$ is upper bounded by $\frac{SIFS+ACK}{CW \cdot S}$. Thus, the probability that the offset between the packets suffices to send an ACK is lower bounded by $1 - \frac{SIFS+ACK}{CW \cdot S}$. For the backward-compatible 802.11g networks, the parameters are $S = 20\ \mu$s, $ACK = 30\ \mu$s, $SIFS = 10\ \mu$s [Gast 2005]. Substituting in the above equations, we find that the success probability is at least 0.9375. ■

There exist, however, patterns that ZigZag can decode but cannot ack synchronously. For example, in Figure 2.4, with a high probability, we can synchronously ACK the first four patterns. However, the last two patterns require asynchronous ACKs. ZigZag always prefers to use synchronous ACKs. Specifically, the AP identifies ZigZag-aware senders during association. It always tries to send synchronous ACKs but if that fails and the sender is ZigZag-aware, the AP sends the ACK asynchronously in a manner similar to [Woo et al. 2007]. In practice, however, most collisions tend to involve two terminals and the autorate algorithm matches the bit rate to the SNR. Thus, we believe that even if the AP does not implement asynchronous ACKs, it can still resolve the majority of the collisions that occur in practice.

2.7 Beyond Two Interferers

Our description, so far, has been limited to a pair of colliding packets. ZigZag, however, can resolve a larger number of colliding senders. Consider the scenario in Figure 2.10, where we have three collisions from three different senders. We refer to the colliding packets by P_1, P_2, and P_3, and collision signals by C_1, C_2, and C_3. The figure shows a possible decoding order. We can start by decoding chunk 1 in the first collision, C_1, and subtract it from C_2 and C_3. As a result, chunk 2 in C_2 becomes interference-free and thus decodable. Next, we subtract chunk 2 from both C_1 and C_3. Now, chunk 3 in C_3 becomes interference-free; so we decode it and subtract it from both C_1 and C_2. Thus, the idea is to find a decoding order such that, at each point, at least one collision has an interference-free chunk ready for decoding.

The following linear-time algorithm provides a chunk-decoding order for any number of collisions.

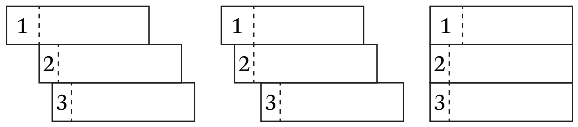


Figure 2.10 **Applying ZigZag to three collisions.**

Step 1. For each of the collisions, decode all the overhanging chunks that are interference-free.

Step 2. Subtract the known chunks wherever they appear in all collisions.

Step 3. Decode all the new chunks that become interference-free as a result of Step 2.

Step 4. Repeat the last two steps until all the chunks from all the packets are decoded.

We would like to estimate how often this linear-time algorithm succeeds in resolving collisions, i.e., the probability that it will not get stuck before fully decoding all symbols. To do so, we simulate the behavior of the 802.11 MAC. Specifically, we have n nodes, all hidden from each other, and all want to transmit a packet at $t = 0$. Each node maintains a congestion window cw, which is initialized to 32 slots. Each node randomly picks a slot in its congestion window to transmit the packet. If a collision occurs and the AP fails to decode the packet, the sender doubles its congestion window, up to a maximum of 1024 slots. The experiment is repeated 10,000 times for each value of n. Figure 2.11 shows the probability that the greedy decoder fails to decode n packets given n collisions. It shows that this probability ranges between .01% and 1%, and hence is negligible in practice.

Intuitively, one may think of the system of n collisions of n packets as a linear system of n equations and n unknowns. The collisions are the linear equations, whereas the packets are the unknowns. Such system is solvable if the equations are linearly independent, i.e., the packets combine differently in different collisions. A general system of linear equations, however, is not always solvable in linear time (it requires a matrix inversion). But the equations in the case of collisions have a special structure because the symbols in a packet appear in all collisions in the same order. Figure 2.11 shows that for such a structure a linear-time decoder is quite powerful. Indeed, for three collisions (or less) we can show that our linear-time algorithm is as powerful as a non-linear decoder. Specifically, we prove the following.

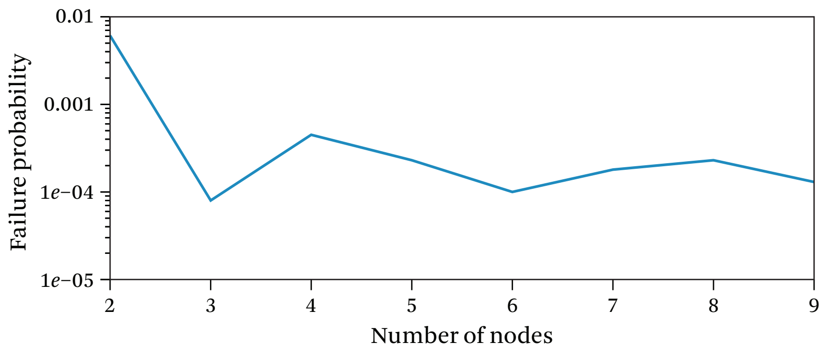


Figure 2.11 **Failure probability of our linear-time decoder as a function of the number of colliding nodes.**

Lemma 2.2 Given three collisions of three packets, if for any packet pair P_i and P_j, there exists two collisions such that this pair has combined differently (in terms of offsets) in these two collisions, the above greedy algorithm always succeeds in decoding all symbols in all colliding packets.

Finally, note that Figure 2.11 is an upper bound on the performance of our linear decoder. In practice, imperfections in the implementation of the decoder limit the maximum number of colliding senders that can be correctly decoded. In Section 2.9.6, we show experimental results for scenarios with three interfering senders.

2.8 Complexity

ZigZag is linear in the number of colliding senders. In comparison to current decoders, ZigZag requires only two parallel decoding lines so that it can decode two chunks in the same time that it would take a current decoder to decode one chunk. Most of the components that ZigZag uses are typical to wireless receivers. ZigZag uses the decoders and the encoders as black boxes. Correlation, tracking, and channel estimation are all typical functionalities in a wireless receiver [Meyr et al. 1998, Castoldi 2002].

2.9 Experimental Environment

We evaluate ZigZag in a 14-node GNURadio testbed. The topology is shown in Figure 2.12. Each node is a commodity PC connected to a USRP GNU radio.[6]

6. http://ettus.com; last retrieved May 2014.

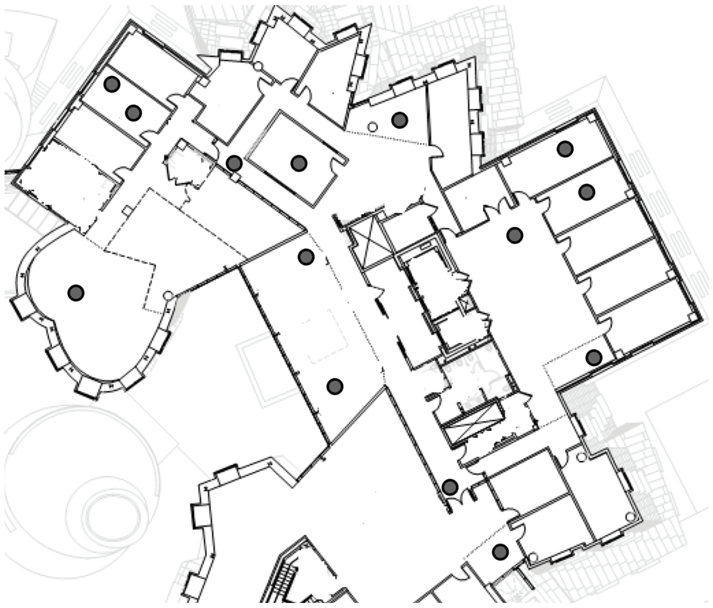


Figure 2.12 Testbed topology.

(a) Hardware and Software Environment. We use the Universal Software Radio Peripheral (USRP) for our RF frontend. We use the RFX2400 daughterboards, which operate in the 2.4 GHz range. The software for the signal processing blocks is from the open source GNURadio project.[7]

(b) Modulation. ZigZag uses the modulation/demodulation module as a black box and works with a variety of modulation schemes. Our implementation, however, uses Binary Phase Shift Keying, $BPSK$, which is the modulation scheme that 802.11 uses at low rates.

(c) Configuration Parameters. We use the default GNURadio configuration, i.e., on the transmitter side, the DAC rate is $128e6$ samples/s, the interpolation rate is 128, and the number of samples per symbol is 2. On the receiver side, the ADC rate is $64e6$ samples/s and the decimation rate is 64. Given the above parameters and a BPSK modulation, the resulting bit rate is 500kb/s. Each packet consists of a 32-bit preamble, a 1500-byte payload, and 32-bit CRC.

7. http://gnuradio.org/redmine/projects/gnuradio; last retrieved May 2014.

(d) Implementation Flow Control. On the sending side, the network interface pushes the packets to the GNU software blocks with no modifications. On the receiving side, the packet is first detected using standard methods built in the GNURadio software package. Second, we try to decode the packet using the standard approach (i.e., using the BPSK decoder in the GNURadio software). If standard decoding fails, we use the algorithm in Section 2.4.1 to detect whether the packet has experienced a collision, and where exactly the colliding packet starts. If a collision is detected, the receiver matches the packet against any recent reception, as explained in Section 2.4.2. If no match is found, the packet is stored in case it helps decoding a future collision. If a match is found, the receiver performs chunk-by-chunk decoding on the two collisions, as explained in Section 2.4.3. Note that even when the standard decoding succeeds, we still check whether we can decode a second packet with lower power (i.e., a capture scenario).

(e) Compared Schemes. We compare the following.

ZigZag. This is a ZigZag receiver as described in Section 2.4 augmented with the backward-decoding described in Section 2.5.

802.11. This approach uses the same underlying decoder as ZigZag but operates over individual packet.

Collision-Free Scheduler. This approach also uses the same basic decoder but prevents interference altogether by scheduling each sender in a different time slot.

(f) Metrics. We employ the following metrics.

Bit Error Rate (BER). The percentage of incorrect bits averaged over every 100 packets.

Packet Loss Rate (PER). This is the percentage of incorrectly received packets. We consider a packet to be correctly received if the BER in that packet is less than 10^{-3}. This is in accordance with typical wireless design, which targets a maximum BER of 10^{-3} before coding (and 10^{-5} after coding) [Intersil Corp. 2000, Tan 2006].[8]

Throughput. This is the number of delivered packets normalized by the GNU Radio transmission rate. Again, a packet is considered delivered if the uncoded BER is less than 10^{-3}. In comparison to packet loss rate, the throughput is more resilient

8. For example, 802.11a target packet error rate (PER) is 0.1 for a packet size of 8000 bits. Given a maximum uncoded BER of 10^{-3}, practical codes like BCH Code(127,99) and BCH Code(15,5) achieve the desired PER.

to hidden terminals in scenarios that exhibit capture effects. This is because the terminal that captures the medium transmits at full rate and gets its packets through, causing unfairness to the other sender, but little impact on the overall throughput.

2.9.1 Setup

Since ZigZag acts exactly like current 802.11 receivers except when a collision occurs, our evaluation focuses on scenarios with hidden terminals, except in Section 2.9.5, where we experiment with various nodes in the testbed irrespective of whether they are hidden terminals. In every run, two (or three) senders transmit 500 packets to an access point. The AP (i.e., the receiver) logs the received signal and the logs are processed offline with the evaluated receiver designs.

Software radios are incapable of accurately timing their carrier sense activity (CSMA) because they perform all signal processing in user mode on the PC. To approximate CSMA, we take the following measures. First, we setup an 802.11a node next to each of our USRP nodes. The objective is to create an 802.11a testbed that matches the topology in our USRP testbed but uses standard 802.11a cards, and copy the results of carrier sense from it to our USRP testbed.

For each USRP experiment, we check whether the corresponding 802.11a nodes can carrier sense each other. Specifically, we make each pair of the 802.11 nodes transmit at full speed to a third node considered as an AP, log the packets, and measure the percentage of packets each of them delivers to the AP. Next, we try to mimic the same behavior using the USRP nodes, where each packet that was delivered in the 802.11 experiments results in a packet delivery in the USRP experiments between the corresponding sender–receiver USRP pairs. Lost 802.11 packets are divided into two categories: collisions and errors. Specifically, a lost 802.11 packet that we can match with a loss from the concurrent sender is considered a collision loss. Other losses are considered as medium errors and ignored. We try to make each USRP experiment match the collisions that occurred in the corresponding 802.11a experiment by triggering as many collisions as observed in the 802.11a traces. The USRP experiments are run without CSMA. Each run matches an 802.11 run between the corresponding nodes. Each sender first transmits the same number of packets that the corresponding 802.11 correctly delivered in the matching 802.11 run. Then both senders transmit together as many packets as there were collisions in the matching 802.11 run.

Software radios also cannot time 802.11 synchronous ACKs. Given the 802.11a traces, we know when a collision occurs, and that the sender should retry the packet, in which case the sender transmits each packet twice. However, if the ZigZag AP manages to decode using a single collision, we ignore the retransmission and do not count it against the throughput. Our prototype does not include the acking scheme described in Section 2.6.

2.9.2 Micro-Evaluation

We examine the role of various components of ZigZag.

(a) Correlation as a Collision Detector. We estimate the effectiveness of the correlation-based algorithm (Section 2.4.1) in detecting the occurrence of collisions. Our implementation sets the threshold to $\Gamma'(\Delta) > \beta \times L \times SNR$, where β is a constant, L is the length of the preamble and SNR is a coarse estimate of the SNR of the colliding sender, which could be obtained from any previously decoded packets or from one of the sender's interference-free chunks. For our testbed, $\beta = 0.6$–0.7 balances false positives with false negatives. Higher values eliminate false positives but make ZigZag miss some collisions, whereas lower values trigger collision detection on clean packets. Note that neither false positives nor false negatives produce end-to-end errors. The harm of false positive is limited to computational resources, because in ZigZag marking a packet as a collision does not prevent correct decoding of that packet. The algorithm behaves as if the packet suffered capture effect and hence is decodable despite being marred by collision. False negatives, on the other hand, make ZigZag miss opportunities for decoding collisions but do not produce incorrect decoding. Our evaluation sets $\beta = 0.65$.

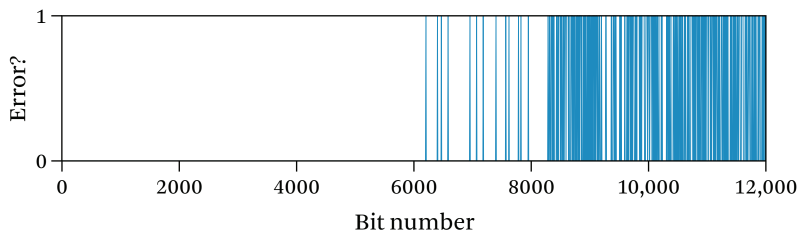
For SNRs in the range 6–20 dB, we run the collision detector on sets of 500 non-collision packets and 500 collisions, and report the results in Table 2-1. The average false positive rate (packets mistaken as collisions) is 3.1%, and the average false negative rate (missing collisions) is 1.9%. Thus, the collision detector is pretty accurate for our purpose.

Table 2.1 Micro-evaluation of ZigZag's components

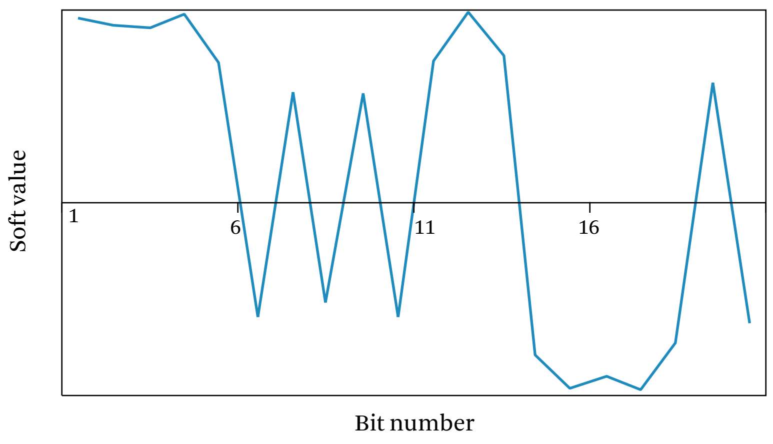
Correlation	False positives	3.1%	
	False negatives	1.9%	
Frequency	Packet size (bytes)	800	1500
and	Success with	99.6%	98.2%
Phase Tracking	Success without	89%	0%
ISI Filter	SNR	10 dB	20 dB
	Success with	99.6%	100%
	Success without	47%	96%

(b) Frequency and Phase Tracking. We evaluate the need for the frequency and phase tracking described in Section 2.4.4b. We disable our tracking algorithm (but leave the decoder unchanged) and provide the encoder with an initially accurate estimate of the frequency offset (as estimated by the decoder). We run ZigZag with and without tracking on 500 collision-pairs of 1500-byte packets. We find that without tracking none of the colliding packets is decodable (BER $> 10^{-3}$), whereas with tracking enabled, 98.2% of the colliding packets are decodable.

Figure 2.13(a) explains this behavior. It plots the error as a function of the bit index in one of the colliding packets (black shades refer to errors). It shows that the first 6000 bits are decoded correctly, but as we go further the bits start getting flipped, and eventually most of the bits are in error. This is expected since even a small residual error in the frequency offset causes a phase rotation that increases linearly with time. Hence, after some time the phase becomes completely wrong causing high decoding



(a) Error distribution due to residual δf



(b) ISI prone symbols

Figure 2.13 Effects of residual frequency offset and ISI.

error rates. This effect is particularly bad for long packets since the errors accumulate over time. Table 2-1 shows that while ZigZag can decode 89% of the 800-byte packets without phase tracking, none of the 1500-byte packets is successfully decoded unless we enable phase tracking.

(c) Effect of ISI. Figure 2.13(b), shows a snapshot of the ISI-affected received bits in our testbed. Recall that BPSK represents a "0" bit with -1 and a "1" bit with $+1$. The figure shows that the value of a received bit depends on the value of its neighboring bits. For example, a "1" bit tends to take a higher positive value if it is preceded by another "1", than if the preceding bit is a "0" bit.

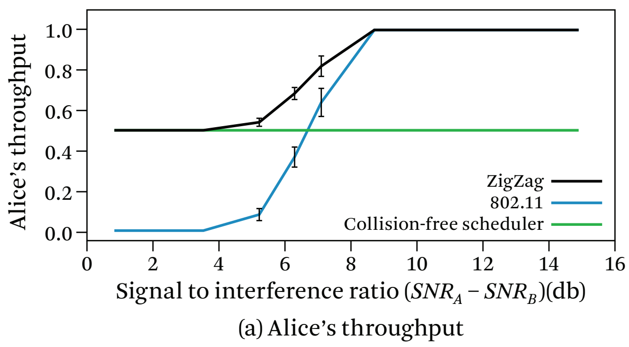
We evaluate the importance of compensating for these distortions using the inverse filter described in Section 2.4.4d. We try to decode 500 collision pairs at different SNRs, with the filter on and off. Table 2.1 shows that, while the filter is not important at high SNRs; i.e., 20 dB, it is necessary at low SNRs. This is expected as at low SNRs, the decoder has to combat both higher noise and ISI distortions.

2.9.3 Does ZigZag Work?

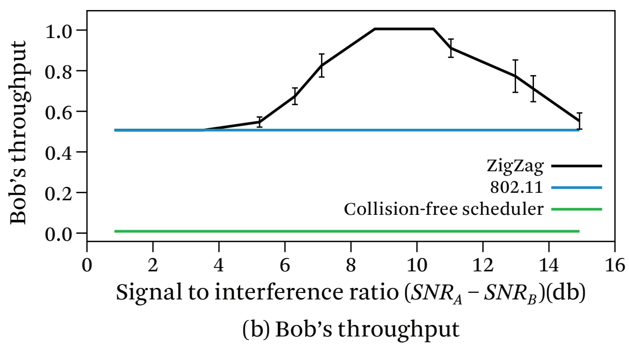
We would like to understand the impact of the signal-to-interference ratio (SINR) on ZigZag's performance. We want to check that ZigZag does not suffer from the same restrictions as traditional interference cancellation, i.e., it works even when the colliding senders have comparable SNRs. We also want to check that ZigZag continues to work as the SNR difference becomes large, i.e., in scenarios that may cause capture effects [Lee et al. 2007, Judd and Steenkiste 2005].

We consider the hidden terminal scenario in Figure 2.1, where Alice and Bob cannot sense each other and hence transmit simultaneously to the AP. We start from a setting where both senders are at equal distance from the AP, i.e., $SNR_A = SNR_B$, and hence $SINR = 0$. Gradually, we move Alice closer to the AP. As Alice moves closer, her SNR at the AP increases with respect to Bob's, making it easier for the AP to capture Alice's signal. We plot the results of this experiment in Figure 2.14, for when the nodes use a collision-free scheduler, 802.11, and ZigZag.

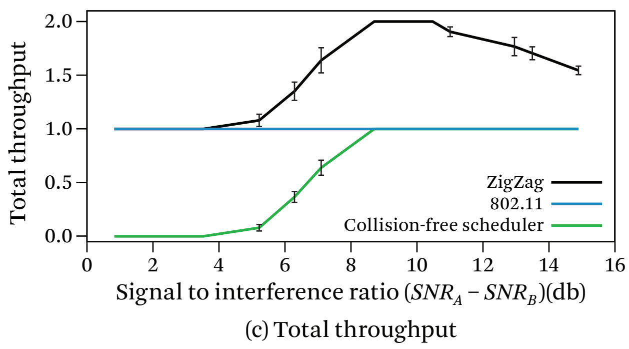
Figure 2.14 shows that ZigZag improves both throughput and fairness. In 802.11, when Alice and Bob are equal distance from the AP, their signals collide, and neither can be received. As Alice moves closer, her signal improves with respect to Bob's. When Alice's signal is 4–6 dB higher than Bob's, the capture effect starts, and we see a slight increase in Alice's throughput. As Alice gets even closer, Bob's signal becomes irrelevant. Note, however, that at all times Bob is never received at the AP with 802.11. In contrast, with the collision-free scheduler, both Alice and Bob get a fair chance at ac-



(a) Alice's throughput



(b) Bob's throughput



(c) Total throughput

Figure 2.14 **Impact of SINR.** The figure plots the throughput of the hidden terminals Alice and Bob, as Alice moves closer to the AP, i.e., as $SINR \approx SNR_A - SNR_B$ increases. It shows that ZigZag achieves higher throughput than both 802.11 and the collision-free scheduler. ZigZag is also fairer than 802.11, where Bob cannot get any packets through.

cessing the AP. But the scheduler cannot exploit that as Alice gets closer, the capacity increases [Tse and Vishwanath 2005], making it possible to decode both Alice and Bob.

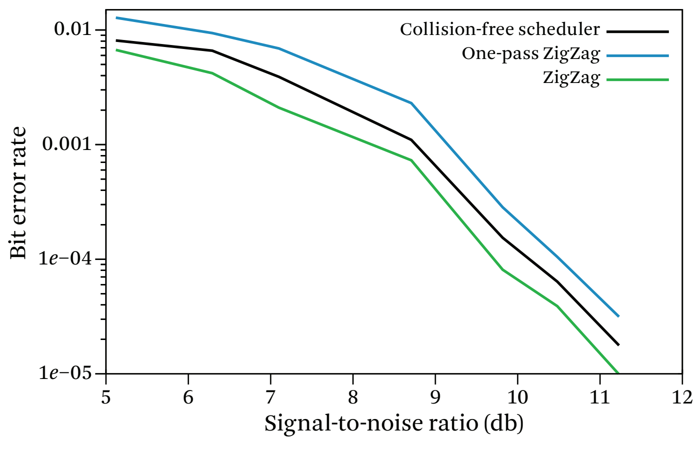
ZigZag outperforms both current 802.11 and the collision-free scheduler. When Alice and Bob are equal distance from the AP, it ensures that they are both received, as if they were allocated different time slots. As Alice moves closer to the AP, the capture effect starts kicking off. As a result, the AP can decode Alice's signal without the need for a second collision. The AP then subtracts Alice's signal from the collision and decode Bob's packet, and thus the total throughput becomes twice as much as the radio transmission rate. As Alice gets even closer, her signal completely covers Bob's signal making it impossible to decode Bob's packet. Thus, this experiment reveals the following.

- At low SINRs, ZigZag significantly outperforms 802.11 and is similar to a collision-free scheduler; i.e., it delivers the same throughput as if the colliding packets were scheduled in separate time slots.
- At high SINR, ZigZag can outperform both 802.11 and the collision-free scheduler. This is because neither 802.11 nor the collision-free scheduler can benefit from scenarios where the network capacity is higher than the sum of the rates of the two senders. In contrast, ZigZag can exploit such scenarios to double the throughput of the network, decoding both hidden terminals using a single collision. Furthermore, ZigZag does not need to be explicitly informed of the capacity of the network to exploit it. It naturally transitions to exploit the increased capacity as the SNR increases.

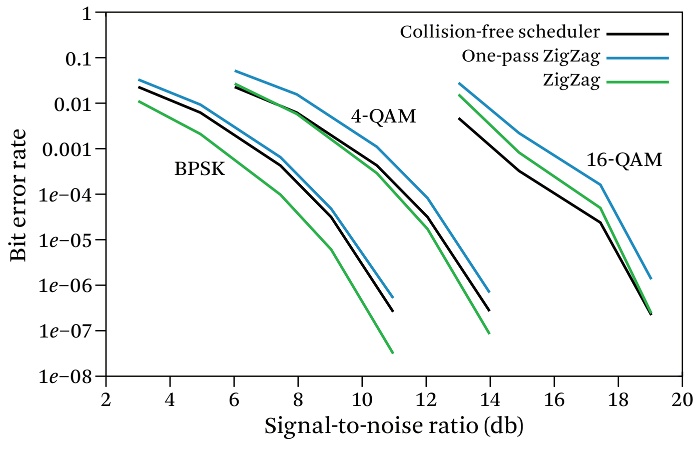
2.9.4 The Impact of the SNR

The standard performance metric for a receiver is the BER as a function of the SNR [Tan 2006, Intersil Corp. 2000, Tse and Vishwanath 2005], and the ultimate test for a design that resolves collisions is whether it can match the uncoded BER of a collision-free reception at every SNR, and for every modulation scheme.

To test performance under various SNRs and modulation schemes, we consider the scenario where Alice and Bob cannot sense each other and hence transmit simultaneously to the AP. In contrast to Section 2.9.3 however, Alice and Bob stay at a fixed and equal distance from the AP. We control their transmission powers to ensure that they have the same SNR, and plot the BER as a function of the SNR. Our GNURadio prototype employs BPSK but to check performance with other modulation schemes (e.g., 4-QAM, 16-QAM), we use simulations. The simulations are based on an additive white Gaussian noise (AWGN) channel [Tse and Vishwanath 2005]. Other parameters (e.g., the packet size and frequency offset) are set to their values in the testbed.



(a) Testbed results



(b) Simulation results

Figure 2.15 **Comparison of Bit Error Rate (BER).** For all modulation schemes, ZigZag and the collision-free scheduler achieve the same BER for comparable SNRs (±1 dB of each other).

Figs. 2.15a and 2.15b plot the BER as a function of the SNR, both in the testbed and in simulations.[9] The plots are only for ZigZag and the collision-free scheduler because,

9. As expected, BPSK in the testbed works at slightly higher SNR than in simulations because of hardware and software imperfections.

in this scenario, 802.11 performed extremely poorly, with BER close to 50%. The figures show the following.

- For all modulation schemes, ZigZag and the collision-free scheduler achieve the same BER for comparable SNRs, i.e., the required SNRs are within 1 dB of each other.
- At BPSK and 4-QAM, ZigZag has a slightly better BER than if the two packets were received collision-free. This is because, in ZigZag, every bit is received twice, once in every collision, improving its chances of being correctly decoded. This impact is countered by error propagation (see Section 2.5). Since errors propagate further in denser modulations, ZigZag's performance is slightly worse at 16-QAM.

2.9.5 Testbed Throughput and Loss Rate

In this section, we use the testbed in Figure 2.12 as a case study to investigate how ZigZag affects various sender–receiver pairs. The testbed has 14 nodes that form a variety of line-of-sight and non-line-of-sight topologies. While up to now we have focused only on scenarios with hidden terminals, in this section we experiment with various testbed nodes irrespective of whether they are hidden terminals. Specifically, we pick two senders randomly. We pick an AP randomly from the nodes reachable by both senders. We mimic CSMA as explained in Section 2.9.1 and make each sender transmit 100 packets to the AP. We repeat the experiment with random set of sender pairs and different choice of APs. Among the sender pairs that we sampled 10% are perfect hidden terminals, 10% can sense each other partially, and 80% can sense each other perfectly.

First, we compare the throughput and loss rate under current 802.11 and ZigZag, for the whole network. Figure 2.16 plots a CDF of the aggregate throughput, i.e., the sum of the throughput of each pair of concurrent senders. The figure shows that in our testbed, ZigZag increases the average throughput by 25.2%. This improvement arises from two factors. For all cases where the normalized aggregate throughput is less than 1, the improvement comes purely from ZigZag' s ability to resolve successive collisions. For cases where the aggregate throughput is higher than 1, the improvement is caused by a combination of being able to resolve a single collision whenever possible, and successive collisions otherwise. Note that traditional interference cancellation applies only to cases whose throughputs are between 1.5 and 2, which are very few. Figure 2.17 plots a CDF of the loss rates of individual sender–receiver pairs, i.e., the flows we experimented with. The figure shows that in our testbed, ZigZag reduces the average packet loss rate from 15.8% to 0.2%.

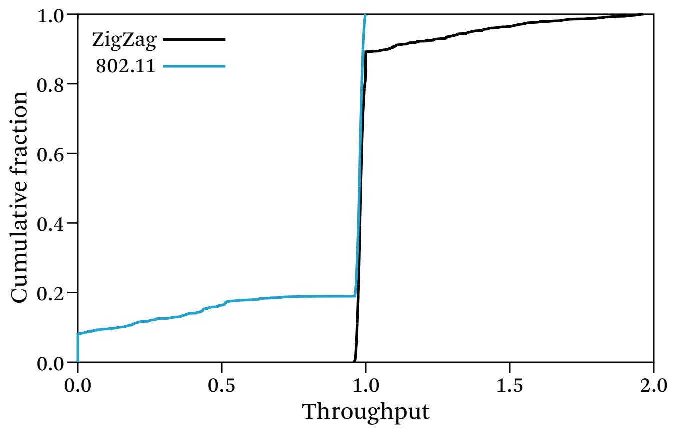


Figure 2.16 **Normalized throughput for the whole testbed.** The figure shows a CDF of the throughputs in our testbed for pairs of competing flows, for both hidden and non-hidden terminal scenarios. ZigZag improves the average throughout in our testbed by 25.2%.

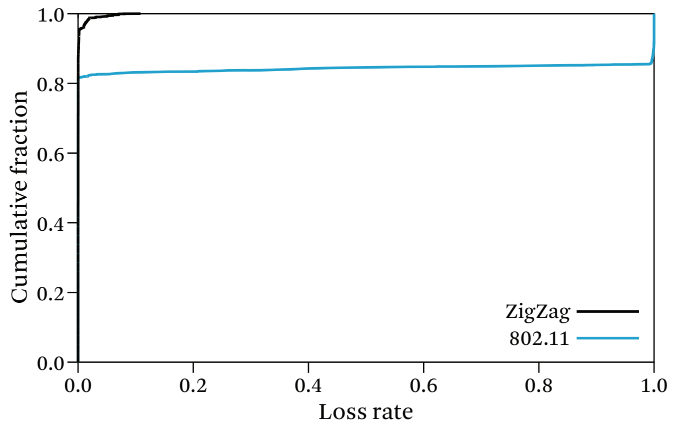


Figure 2.17 **Loss rate for the whole testbed.** The figure shows a CDF of the packet loss rate in our testbed for pairs of competing flows, for both hidden and non-hidden terminal scenarios. ZigZag improves the average loss rate in our testbed from 15.8% to 0.2%.

Next, we check that a ZigZag AP is always a conservative choice and does not hurt any flow. Figure 2.18 shows a scatter plot of the throughout of every sender–receiver pair in our experiments, both under 802.11 and ZigZag. The figure shows that ZigZag consistently improves the throughput and does not hurt any sender–receiver pair.

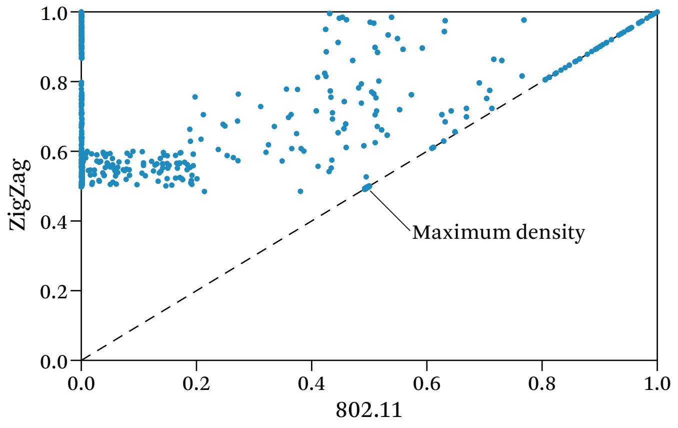


Figure 2.18 **Scatter plot of flow throughputs.** The figure shows a scatter plot of ZigZag and 802.11 throughputs for each sampled sender–receiver pairs. ZigZag helps when there are hidden terminals and never hurts.

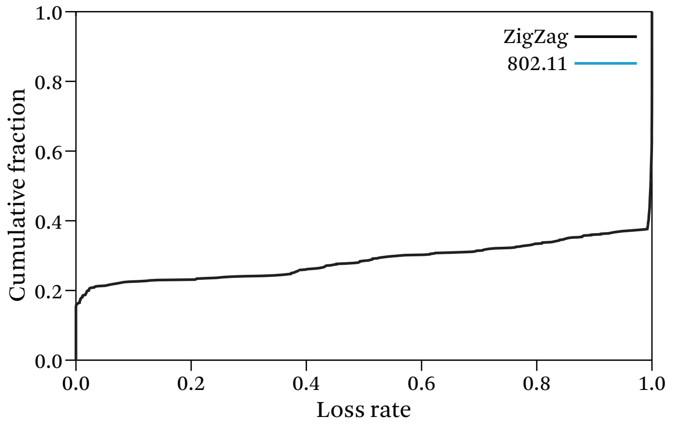


Figure 2.19 **CDF of loss rate at hidden terminals.** The figure zooms on scenarios with full or partial hidden terminals. ZigZag reduces the average loss rate for hidden terminals in our testbed from 72.6% to about 0.7%.

Next, we zoom on the hidden terminals in our testbed, which we define as sender pairs that fail to sense each other fully or partially. Figure 2.19 shows a CDF of the packet loss rate in transfers that suffered such hidden terminal scenarios. The figure shows that ZigZag improves the average loss rate for hidden terminals in our testbed from

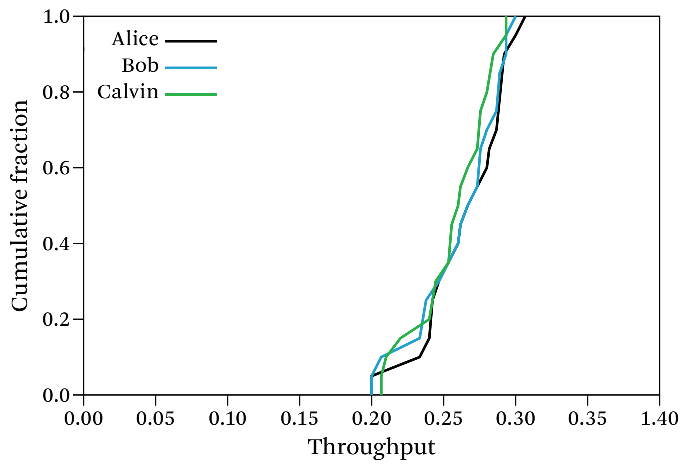


Figure 2.20 **ZigZag's performance with three hidden terminals.** Cumulative distribution of the throughput of three hidden terminals.

72.6% to 0.7%. Further, for some severe cases, the packet loss rate goes down from 99–100% to about 0.

2.9.6 Many Hidden Terminals

In Section 2.7, we generalized ZigZag to deal with many colliding sources. Here, we evaluate how ZigZag performs on three collisions. In this experiment, we have three hidden terminals that transmit concurrently to a random AP. Figure 2.20 shows the CDF of the throughput under ZigZag. The figure shows that all three senders see a fair throughput that is about one third of the medium throughput. Thus, even with more than a pair of colliding senders, ZigZag performs almost as if each of the senders transmitted in a separate time slot.

2.10 Discussion

This chapter presents ZigZag, a receiver that can decode collisions. Our core contribution is a new form of interference cancellation that iteratively decodes strategically picked chunks, exploiting asynchrony across successive collisions. We show via a prototype implementation and testbed evaluation that ZigZag addresses the hidden terminal problem in WLANs, improving the throughput and loss rate.

ZigZag has wider implications for wireless system design than explored in this chapter. It motivates a more aggressive medium access protocols (MAC) that exploits concurrent transmissions in order to increase spatial reuse and network throughput. Further, ZigZag can be combined with ideas in network coding into one system that improves concurrency, addresses hidden terminals, and collects network coding gains.

Beyond these gains, ZigZag introduced a new approach to deal with 802.11 collisions. Traditionally, networking researchers have considered wireless collisions to be a harmless phenomenon and designed mechanisms to avoid them. ZigZag flips this conventional wisdom on its head, and transforms collisions from an intrinsically harmful phenomena to a harmless one. Since ZigZag was published, our approach has been applied to redesign MAC protocols for single-antenna [Li et al. 2011, Sen et al. 2010, Li et al. 2010] and multi-antenna radios [Lin et al. 2011, Shen et al. 2012], to design radios that can transmit and receive on the same channel, i.e., full-duplex radios [Choi et al. 2010, Duarte and Sabharwal 2010], and also to reduce the power consumption of low-power RFID devices [Wang et al. 2012].

3 Combating High-Power Cross-Technology Interference

Cross-technology interference is emerging as a major problem for 802.11 networks. Independent studies in 2010 by the Farpoint Group [Farpoint Group 2010], BandSpeed [Bandspeed 2010], and Miercom [Miercom 2010] all show that high-power interferers like baby monitors and cordless phones can cause 802.11n networks to experience a complete loss of connectivity. Other studies from Ofcom [Ofcom 2009], Jupiter Research [Cisco 2007], and Cisco [Cisco 2010] report that such interferers are responsible for more than half of the problems reported in customer networks. Today's high-power non-WiFi sources in the ISM band include surveillance cameras, baby monitors, microwave ovens, digital and analog cordless phones, and outdoor microwave links. Some of these technologies transmit in a frequency band as wide as 802.11, and all of them emit power that is comparable or higher than 802.11 devices [Bandspeed 2010]. Further, the number and diversity of such interferers is likely to increase over time due to the proliferation of new technologies in the ISM band.

Traditional solutions that increase resilience to interference by making 802.11 fall down to a lower bit rate are ineffective against high-power cross-technology interference. As a result, the most common solution today is to hop away to an 802.11 channel that does not suffer from interference[1] [Yang et al. 2010, Moscibroda et al. 2008, Rahul et al. 2008]. However, the ISM band is becoming increasingly crowded, making it difficult to find an interference-free channel. The lack of interference-free channels has led WiFi device manufacturers[2, 3] and researchers [Lakshminarayanan et al. 2009] to develop signal classifiers that inform the 802.11 user about the root cause of the

1. http://www.cisco.com/en/US/netsol/ns1070/index.html; last retrieved May 2014.
2. http://www.bandspeed.com; last retrieved May 2014.
3. http://www.airdefense.net; last retrieved May 2014.

problem (e.g., Bluetooth, microwave, baby monitor). However, these classifiers put the burden of addressing the problem on the user and cannot solve the problem on their own.

In this book, we ask whether it is possible to use the MIMO capability inherent to 802.11n to address high-power cross-technology interference. MIMO achieves most of its throughput gains by enabling multiple concurrent streams (e.g., packets). Current MIMO decoding, however, fails if any of these concurrent streams belongs to a different technology. Nonetheless, if MIMO can be made to work across technologies, a 3×3 802.11n transmitter can then treat the signal from a baby monitor or microwave as one stream and still deliver two concurrent streams to its receiver.

The challenge in harnessing MIMO across different technologies stems from the fact that MIMO decoding hinges on estimating the channel between all transmit and receive antennas. These estimates rely on understanding the signal structure and assume a known preamble. Hence, it has been infeasible to use MIMO across different and potentially unknown technologies.

We present TIMO,[4] an 802.11n receiver design robust to high-power cross-technology interference. TIMO introduces a MIMO technique that enables a receiver to decode a signal of interest, even when the channel from other concurrent transmissions is unknown. The intuition underlying TIMO is best explained via an example. Consider a pair of two-antenna 802.11n nodes that want to communicate in the presence of a high-power unknown interferer. Let $s(t)$ be the signal of interest and $i(t)$ the interference signal. The 802.11n receiver node will receive the following signals on its two antennas:[5]

$$y_1(t) = h_i i(t) + h_s s(t) \tag{3.1}$$

$$y_2(t) = h_i' i(t) + h_s' s(t), \tag{3.2}$$

where h_i and h_i' are the channels from the interferer to the 802.11n receiver and h_s and h_s' are the channels from the 802.11n sender to the 802.11n receiver. The 802.11n receiver has to solve these equations to obtain its signal of interest $s(t)$. It knows the received samples, $y_1(t)$ and $y_2(t)$, and the channels from its transmitter, h_s and h_s', which can be computed in the presence of interference (see Section 3.5.4). The receiver, however, cannot compute the channels from the interferer, h_i and h_i', because it does not

4. Technology Independent Multi-Output (TIMO) receiver design.

5. The equations here are for single-tap channels. Subsequent sections extend these equations to multi-tap channels.

know the interferer's signal structure or preamble. Hence, it is left with two equations in three unknowns ($s(t)$, $h_i i(t)$, and $h'_i i(t)$),[6] which it cannot solve.

Note that the receiver can cancel the interference if it knows the interferer's channel ratio $\frac{h_i}{h'_i}$. In particular, the receiver can rewrite Equations 3.1 and 3.2 to express the signal of interest as

$$s(t) = \frac{y_1(t) - \beta y_2(t)}{h_s - \beta h'_s} \quad \text{for } \beta = \frac{h_i}{h'_i}.$$

The only unknown in the above equation is β. Thus, though the 802.11n receiver cannot compute the exact channels of the interferer, it can still cancel its interference using only its channel ratio.

Still, how do we obtain this ratio given no support from the interferer? The receiver can obtain this ratio as follows: Suppose that for some time instance $t = t_0$, our transmitter sends a known symbol $s(t_0)$. Our receiver can then substitute in equations 3.1 and 3.2 to obtain

$$\frac{h_i}{h'_i} = \frac{y_1(t_0) - h_s s(t_0)}{y_2(t_0) - h'_s s(t_0)},$$

where all terms are known except for the ratio $\frac{h_i}{h'_i}$. In Section 3.5, we develop this idea further and eliminate the need for having the transmitter send a known symbol, which makes the scheme applicable to existing 802.11n frames. We further generalize the solution to address scenarios in which different frequencies have different interferers, or the interferer hops across frequencies.

A MIMO transmitter can also encode its signal using interference nulling [Tse and Vishwanath 2005] so that it does not interfere with a concurrent transmission from a competing technology. However, using a similar computation, we show that it is necessary to obtain the ratio $\frac{h_{s1}}{h_{s2}}$, where h_{s1} and h_{s2} are the channels from the MIMO transmitter to the receiver of the competing technology. These channels can only be estimated if the receiving node transmits data at some point, i.e., if the competing technology uses bidirectional communication, e.g., a cordless phone. If this constraint is met, however, TIMO can be used not only to protect 802.11n networks from high-power interference, but also as a cognitive mechanism that enables MIMO-based nodes to peacefully coexist in the same frequency band with bidirectional non-MIMO nodes from a different technology. In this case, the simpler non-MIMO nodes just transmit bidirectionally, and the more complex MIMO nodes take on the burden of preventing interference.

6. We can lump $i(t)$ with the channel variable because we are not interested in decoding the symbols of the interferer.

This approach can lead to a new form of spectrum sharing in which different technologies do not necessarily have to find unoccupied bands and, in crowded environments, could instead occupy the same band thereby increasing spectral efficiency.

We have built a prototype of TIMO using two-antenna USRP2 radios. We have evaluated our design in the presence of interference from three technologies: a microwave oven, an analog baby monitor, and a DSSS cordless phone. We first use commercial 802.11n cards and iperf[7] to transmit in the presence of these interferers. We find that, in our testbed, the cordless phone and the baby monitor prevent 802.11 from establishing any connection, reducing its throughput to zero. The microwave, on the other hand, results in a throughput reduction of 35–90%. We replace the commercial 802.11n cards with our USRP2 nodes and repeat the experiment with and without TIMO. We find that in the absence of TIMO, when the USRP2 nodes are less than 31 ft away from the cordless phone or the baby monitors, they cannot deliver any packets. In contrast, in the presence of TIMO, and for the same locations, their throughput increases to 13–23 Mb/s. We also implement cross-technology interference nulling and show that it enables a MIMO node to significantly reduce the packet loss at the receiver of a competing technology, with the reduction in packet loss being as high as 14× in some locations.

3.1 Impact of Cross-Technology Interference on 802.11n

We study the interaction between high-power interferers and 802.11n and compare against the interaction between a low power interferer, Bluetooth, and 802.11n. We focus on three high-power technologies that are prevalent in today's environments [Ofcom 2009]: DSSS cordless phones, baby monitors, and microwave ovens.

Experimental Setup. We use the Netgear N-300 USB-adapter and the Netgear N-300 router as the 802.11n client and AP, respectively. Both devices support 2 × 2 MIMO. We place the AP and the client at positions A and B in Figure 3.1. In each run, we place the interferer at one of the marked locations in Figure 3.1. Our experiments include line-of-sight and non-line-of-sight situations, and show scenarios in which the interferer is within 1 foot of the 802.11n client as well as 90 feet away from it. We run iperf on the two 802.11n devices with the 802.11n client acting as the iperf server. The AP sends UDP packets for 2 minutes and logs the average throughput observed every 500 ms. In each location, we compute the observed 802.11n throughput first when the interferer is turned OFF and next when it is ON.

7. http://iperf.sourceforge.net; last retrieved May 2014.

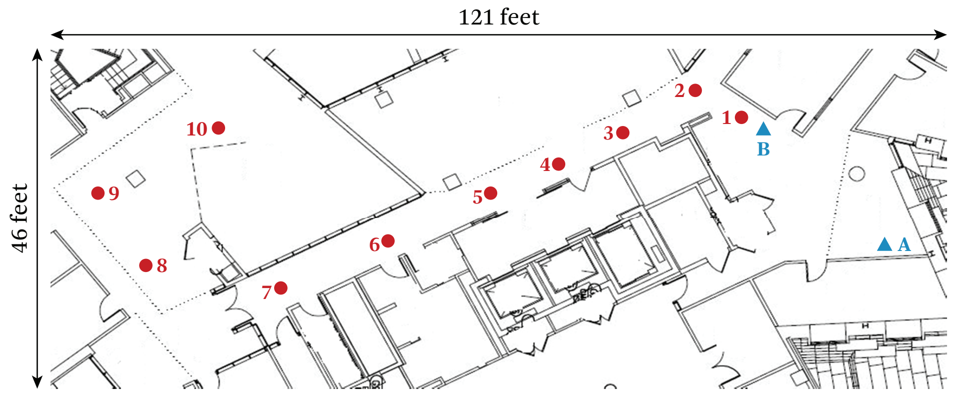


Figure 3.1 **Testbed.** An 802.11n transmitter located at A is communicating with an 802.11n receiver at B. The interferer is placed in one of the locations 1 to 10.

3.1.1 Digital Cordless Phone

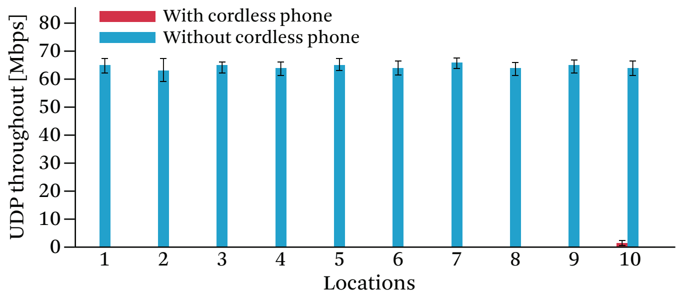
We experiment with the Uniden TRU 4465-2 DSSS cordless handset system. The phone base and handset communicate using digital spread spectrum in the 2.4 GHz range. In each experiment, we fix the 802.11n AP and client at locations A and B and place both the cordless handset and the phone base at one of the locations in the testbed, 5 cm away from each other.

Figure 3.2(a) shows the 802.11n throughput with and without interference from the cordless phone. The figure shows that in the presence of the cordless phone, the 802.11n client and AP could not establish a connection and hence experienced zero throughput.

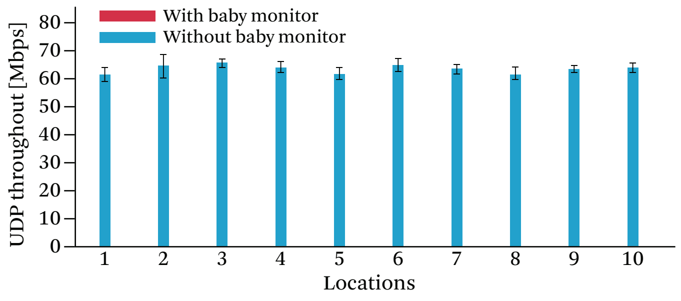
The reason for this the phone base and handset use Time-Division Duplexing (TDD) to communicate in the same frequency band. The handset transmits in the first time slot, followed immediately by a transmission from the phone base. Since these devices continuously transmit, the channel is never free. Thus, an 802.11n node that carrier senses the medium never gets the opportunity to transmit. Furthermore, since the phone transmits at about 25 mW, which is comparable to an 802.11 laptop, its interference continues even at distances as far as 90 ft.

3.1.2 Baby Monitor

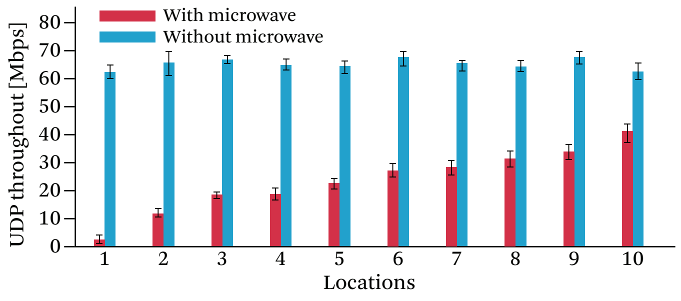
We experiment with the C-501 wireless monitoring toolkit, which has two units: a 2.4 GHz wireless camera that supports up to four different channels (i.e., 2.414 GHz, 2.432 GHz, 2.450 GHz, and 2.468 GHz), and a wireless video receiver. For every interferer location, we measure the 802.11n throughput with the camera ON and OFF, and plot



(a) DSSS phone



(b) Baby monitor



(c) Microwave oven

Figure 3.2 **WiFi throughput in the presence of high-power interferers.**

the results in Figure 3.2(b). The figure shows that the 802.11n client and AP could not establish a connection and, hence, could not exchange any packets for all tested locations.

The reason again is that the camera transmits continuously, thus hogging the medium completely. These observations, compounded with the fact that the camera occu-

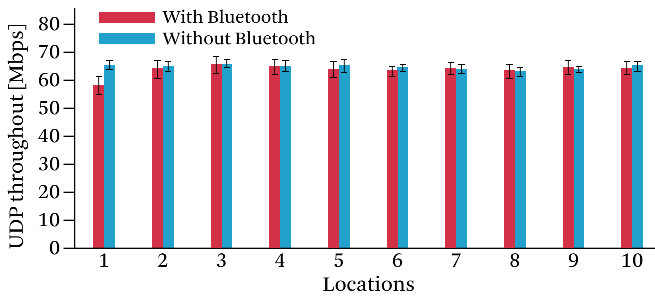


Figure 3.3 **The impact of Bluetooth interference on 802.11n.**

pies a relatively wide channel of 16 MHz and transmits at a fairly high power of 200 mW[8] explain the inability of 802.11n to obtain any throughput.

3.1.3 Microwave Ovens

We use the SHARP R-310CW microwave oven. Figure 3.2(c) shows the observed 802.11n average throughput for different placements of the microwave. The figure shows that when the microwave is 1 foot away (in location 1), 802.11n suffers a throughput reduction of 90%. The 802.11n throughput improves as the microwave is moved away from the AP and its client, and the throughput loss decreases to 35% at the farthest location from the client. While the results are slightly better with a microwave oven, WiFi still experiences significant reduction in throughput across all the locations.

3.1.4 Frequency Hopping Bluetooth

Finally, we evaluate the interference generated by Bluetooth devices. Bluetooth uses frequency hopping across a 79 MHz band in the 2.402–2.480 GHz range, occupying 1 MHz at any point in time. The most common devices use class 2 Bluetooth which transmits at a relatively low power of 2.5 mW.[9]

For each interferer location, we transfer a 100 MB file between two Google Nexus One phones. We plot in Figure 3.3 the throughput obtained by our 802.11n devices, in the presence and absence of the Bluetooth traffic. The figure shows that except in location 1, which is 1 foot away from the 802.11n client, the Bluetooth exchange has no observable impact on the throughput of the 802.11n devices.

8. http://www.genica.com; last retrieved May 2014.

9. http://www.bluetooth.com; last retrieved May 2014.

3.1.5 Summary

The above empirical study shows the following.

- High-power cross-technology interference can completely throttle 802.11n. Furthermore, loss of connectivity can occur even when the interferer is in a non-line-of-sight position and separated by 90 feet.
- While 802.11 and low-power interferers (e.g., Bluetooth) have managed a form of coexistence where both devices stay operational, coexistence with high-power devices (e.g., cordless phones, baby monitors, microwave, etc.) is lacking. Furthermore, the typical outcome of the interaction between 802.11n and a high-power interferer is that 802.11n either suffers a complete loss of connectivity or a significant throughput reduction. In Section 3.8 we show that even if carrier sense is deactivated, 802.11n continues to lose connectivity for many of the interferer's locations.
- Frequency isolation is increasingly difficult. Multiple of the studied interferers occupy relatively wideband channels of 16–25 MHz (e.g., camera and microwave). Moreover, these devices can occupy any band in the 802.11 spectrum. For example, both the cordless phone and the baby monitor have multiple channels that together cover almost the whole frequency range of 802.11.
- Finally, the characteristics of an interferer may change in time and frequency. The interferer may have ON–OFF periods, may move from one frequency to another, or change the width of the channel it occupies, like a microwave. This emphasizes the need for an agile solution that can quickly adapt to changes in the interference signal.

3.2 MIMO and OFDM Background

Consider the 2×2 MIMO system in Figure 3.4. Suppose the sender transmits stream $s_1(t)$ on the first antenna, and $s_2(t)$ on the second antenna. The wireless channel linearly combines the signal samples corresponding to the two streams. Therefore, the receiver receives the following linear combinations on its two antennas:

$$y_1(t) = h_{11}s_1(t) + h_{21}s_2(t) \tag{3.3}$$

$$y_2(t) = h_{12}s_1(t) + h_{22}s_2(t), \tag{3.4}$$

where h_{ij} is a complex number whose magnitude and angle refer to the attenuation and delay along the path from the ith antenna on the sender to the jth antenna on the receiver, as shown in Figure 3.4. If the receiver knows the channel coefficients, h_{ij}, it

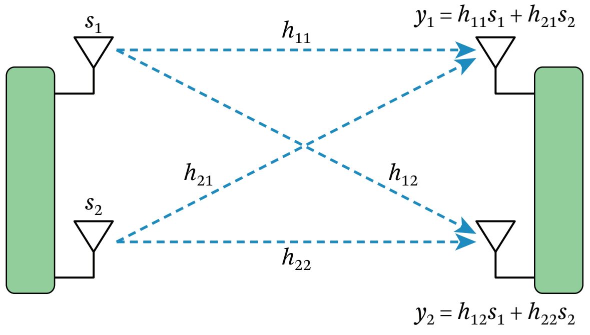


Figure 3.4 **Decoding in a standard 2×2 MIMO system.**

can solve the above two linear equations to obtain the two unknowns, $s_1(t)$ and $s_2(t)$, and decode the two transmitted streams.

To enable the receiver to estimate the channel coefficients, h_{ij}, a MIMO sender starts each frame by transmitting a known preamble from each of its antennas, one after the other. The receiver uses its knowledge of the transmitted preamble and the received signal samples to compute the channel coefficients, which it uses to decode the rest of the bits in the frame.

The above model assumes a narrowband channel, whose bandwidth is limited to a few MHz. In wideband channels, different frequencies may experience different channels. Thus, the channel function cannot be expressed as a single complex number; it has to be expressed as a complex filter, and the multiplication becomes a convolution:

$$y_1(t) = \mathbf{h}_{11} * s_1(t) + \mathbf{h}_{21} * s_2(t)$$
$$y_2(t) = \mathbf{h}_{12} * s_1(t) + \mathbf{h}_{22} * s_2(t).$$

Modern wireless technologies like 802.11a/g/n, WiMax, and LTE handle such wide channels by operating on the signal in the frequency domain using OFDM. OFDM divides the channel frequency spectrum into many narrow subbands called OFDM subcarriers. The receiver takes an FFT of the received signal and operates on individual OFDM subcarriers, as if they were narrowband channels, i.e., the receiver applies the model in Equations 3.3 and 3.4 to the frequency domain signal, and decodes the transmitted symbols.

In 802.11, there are 64 OFDM subcarriers, 4 of which are called pilots that have a known symbol pattern to allow the receiver track the channel [Heiskala and Terry 2001]. Additionally, 48 subcarriers are used to transmit data and the rest are unused for distortion reasons.

3.3 Problem Domain

TIMO deals with high-power cross-technology interference in 802.11n networks. We focus on typical situations that arise in the operation of 802.11 networks. In particular, we have the following.

- TIMO tackles scenarios in which the interferer is a single antenna device. This is typically the case for current 802.11 interferers, like baby monitors, microwave ovens, cordless phones, surveillance cameras, etc.
- TIMO applies to scenarios in which the interfering signal lasts more than a few seconds. This constraint does not necessarily mean that the interferer transmits continuously for that duration. For example, a microwave signal that lasts for a few seconds satisfies our constraint despite having OFF periods.
- TIMO applies to scenarios where, in the absence of an interferer, the 802.11n receiver can use MIMO multiplexing, i.e., it can receive multiple concurrent streams at some bitrate. If the 802.11n receiver cannot multiplex streams from the same technology, it cannot be made to multiplex streams from different technologies.
- TIMO can address environments with multiple concurrent interferers, as long as the interferers are in different frequencies (i.e., different 802.11 OFDM subcarriers). We believe this to be the common case in today's networks because the presence of multiple high-power interferers in the same band will cause them to interfere with each other, and is likely to prevent the proper operation of the device.

3.4 TIMO

TIMO extends the MIMO design to operate across diverse wireless technologies that may differ in modulation, coding, packet format, etc. It develops two primitives: The first primitive enables a MIMO 802.11n pair to exchange packets in the presence of an unknown interference signal, as if the unknown interference were a single-antenna 802.11 transmission. For example, an 802.11n AP-Client pair may use this primitive to correctly decode packets in the presence of the ON periods of a microwave oven. The second primitive enables a MIMO node to transmit in the presence of an unknown bi-directional technology without hampering reception at the receiver of the unknown technology. For example, an 802.11n node may use this primitive to transmit in the presence of a cordless phone without hampering the phone's operation. The next few sections describe this in detail.

3.5 Decoding in the Presence of Cross-Technology Interference

Consider a scenario in which two 802.11n nodes want to communicate in the presence of high-power cross-technology interference. For clarity, we will explain the design in the context of a two-antenna 802.11n receiver decoding a single 802.11n transmission, in the presence of an interferer. The results extend to any number of antennas as we show in [Gollakota et al. 2011].

In this case, the signal at the two-antenna 802.11n receiver is the sum of the signal of interest, $s(t)$, and the interference signal, $i(t)$, after convolving them with their respective channels to the receiver:

$$y_1(t) = \mathbf{h}_i * i(t) + \mathbf{h}_s * s(t) \tag{3.5}$$

$$y_2(t) = \mathbf{h}'_i * i(t) + \mathbf{h}'_s * s(t), \tag{3.6}$$

where $\mathbf{h}_i$ and $\mathbf{h}'_i$ are the channel functions of the interference signal and $\mathbf{h}_s$ and $\mathbf{h}'_s$ are channel functions of the signal of interest. We will explain TIMO's decoding algorithm assuming the receiver knows the channel of the signal of interest. In Section 3.5.4, we explain how the receiver obtains this channel in the presence of interference.

Since the signal of interest (i.e., that of 802.11n) is an OFDM signal, the receiver processes its input in the frequency domain by taking an FFT. Thus, for each OFDM subcarrier, j, the receiver obtains the following equations:

$$Y_{1j} = H_{ij} I_j + H_{sj} S_j \tag{3.7}$$

$$Y_{2j} = H'_{ij} I_j + H'_{sj} S_j, \tag{3.8}$$

where the terms in the above equations are the frequency version of the terms in Equations 3.5 and 3.6, for a particular OFDM subcarrier. Thus, the receiver can express the signal of interest as

$$S_j = \frac{Y_{1j} - \beta_j Y_{2j}}{H_{sj} - \beta_j H'_{sj}} \quad \text{for } \beta_j = \frac{H_{ij}}{H'_{ij}}. \tag{3.9}$$

All terms in Equation 3.9 are known at the receiver, except for β_j. The objective is to figure out β_j in each subcarrier, and use it to decode the signal of interest, S_j, in that subcarrier.

A TIMO receiver has three main components shown in Figure 3.5. (1) An algorithm for computing the interferer's channel ratio in an OFDM subcarrier without knowing the interferer's preamble or signal structure. (2) A decoder that allows the receiver to decode the signal of interest given the interferer's channel ratio in every OFDM subcarrier. (3) An iteration mechanism that reduces the noise in the computation of channel ratios, hence increasing SNR. The following sections describe these components.

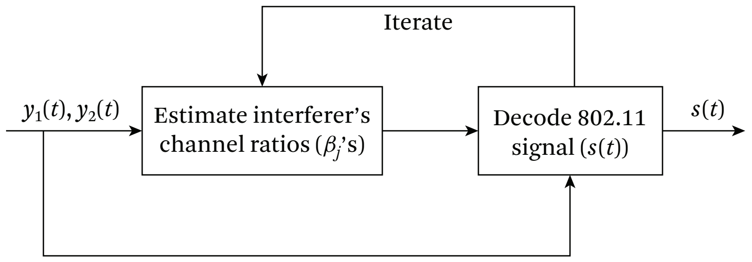

Iterate
$y_1(t), y_2(t)$
Estimate interferer's channel ratios (β_j's)
Decode 802.11 signal ($s(t)$)
$s(t)$

Figure 3.5 Flowchart of the different components.

3.5.1 Computing the Interferer's Channel Ratio

A simplistic approach for computing the ratio $\beta_j = \frac{H_{ij}}{H'_{ij}}$ would rely on that the signal S_j in the OFDM pilots is known to the receiver. Thus, if one assumes β_j is the same for all OFDM subcarriers, one can simply substitute the signal S_j, where j is a pilot subcarrier, in Equation 3.9, and use that equation to compute the ratio β. The receiver then uses this ratio to compute signal values in other OFDM subcarriers that contain data symbols. However, the assumption that the interferer channel ratio is the same in all OFDM subcarriers is typically invalid for several reasons. First, there might be multiple interferers each of them operating in a different frequency band. For example, the interfering signal may be a combination of two cordless phone signals each occupying upto 4 MHz and overlapping with a different set of 802.11n OFDM subcarriers. Second, there might be an interferer that hops across the OFDM subcarriers, but does not always occupy all subcarriers. This is the case for the narrowband signal during the microwave ON period. Finally, the interferer may have a relatively wideband channel, like the baby monitor which can span upto 16 MHz. In this case, the channel of the interferer may differ across the OFDM subcarriers due to multipath and hence the channel ratio also changes across the subcarriers.

Thus, the receiver should compute the interferer's channel ratio for each OFDM subcarrier independently. Since most OFDM subcarriers carry data and contain no known patterns, the receiver has to compute this ratio without any known symbols.

Below we use Equations 3.7 and 3.8 to obtain a closed form expression for the interferer's channel ratio in each OFDM subcarrier. To do so, we first eliminate the contribution from the signal of interest S_j, by multiplying Equation 3.8 with $\frac{H_{sj}}{H'_{sj}}$ and subtracting it from Equation 3.7:

$$Y_{1j} - \frac{H_{sj}}{H'_{sj}} Y_{2j} = \left(\frac{H_{ij}}{H'_{ij}} - \frac{H_{sj}}{H'_{sj}}\right) H'_{ij} I_j.$$

Next, we multiply the above equation with the conjugate of Y_{2j}, and take the expectation:

$$
\begin{aligned}
E[(Y_{1j} - \frac{H_{sj}}{H'_{sj}}Y_{2j})Y^*_{2j}] &= (\frac{H_{ij}}{H'_{ij}} - \frac{H_{sj}}{H'_{sj}})E[H'_{ij}I_jY^*_{2j}] \\
&= (\frac{H_{ij}}{H'_{ij}} - \frac{H_{sj}}{H'_{sj}})E[H'_{ij}I_j(H'^*_{ij}I^*_j + H'^*_{sj}S^*_j)] \\
&= (\frac{H_{ij}}{H'_{ij}} - \frac{H_{sj}}{H'_{sj}})(E[|H'_{ij}I_j|^2] + H'^*_{sj}H'_{ij}E[I_jS^*_j]) \\
&= (\frac{H_{ij}}{H'_{ij}} - \frac{H_{sj}}{H'_{sj}})E[|H'_{ij}I_j|^2] \\
&= (\beta_j - \frac{H_{sj}}{H'_{sj}})P'_{Ij},
\end{aligned}
\tag{3.10}
$$

where $|x|^2 = xx^*$ denotes the square of the amplitude of the complex number x and $E[I_jS^*_j] = 0$ because the signal of interest is independent from the interference signal and hence their correlation is zero. Also, $P'_{Ij} = E[|H'_{ij}I_j|^2]$ is the received interference power in OFDM subcarrier j on the second antenna of the 802.11n receiver.

Equation 3.10 has two unknown β_j and P'_{Ij}. Thus, if the receiver knows the interferer's received power, P'_{Ij}, it can solve Equation 3.10 to obtain the desired ratio. To compute P'_{Ij}, the receiver takes Equation 3.8, multiplies it by its conjugate, and then computes the expectation:

$$
\begin{aligned}
E[Y_{2j}Y^*_{2j}] &= E[(H'_{ij}I_j + H'_sS_j)(H'_{ij}I_j + H'_sS_j)^*] \\
&= E[|H'_{ij}I_j|^2] + E[|H'_sS_j|^2] \\
&= P'_{Ij} + P'_{Sj},
\end{aligned}
\tag{3.11}
$$

where P'_{Sj} is the power of the signal of interest on the second antenna in the jth OFDM subcarrier. Again, to reach Equation 3.11 we have exploited the fact that the interference signal and the signal of interest are independent of each other.

We can solve Equations 3.10 and 3.11 together to obtain the ratio

$$
\beta_j = \frac{H_{ij}}{H'_{ij}} = \frac{E[(Y_{1j} - \frac{H_{sj}}{H'_{sj}}Y_{2j})Y^*_{2j}]}{E[|Y_{2j}|^2] - P'_{Sj}} + \frac{H_{sj}}{H'_{sj}}.
\tag{3.12}
$$

This equation enables the 802.11 receiver to compute the interferer's channel ratio without any known symbols, simply by substituting the power and the channel ratio for $s(t)$.

It is important to note that the above derivation exploits that expectations can be computed by taking averages. The accuracy of this estimate increases as one averages over more signal symbols. In Section 3.5.3 we will discuss how we can obtain a good accuracy without averaging over many symbols.

3.5.2 Decoding the Signal of Interest

Once the 802.11n receiver has an estimate of the interferer's channel ratio, β_j, in each OFDM subcarrier, it proceeds to decode its own signal of interest. One way to decode would be to substitute β_j in Equation 3.9 to compute S_j in the frequency domain. This approach works well when the interferer is a narrowband signal, like a cordless phone. However, it has low accuracy in scenarios the interferer has a relatively wideband channel, like a baby monitor that spans 16 MHz. This is because wideband signals suffer from multipath effects, i.e., the signal travels from the sender to the receiver along multiple paths with different delays. A wideband receiver receives the combination of multiple copies of the same signal with different relative delays. This leads to inter-symbol interference (ISI), which mathematically is equivalent to convolving the time-domain signal with the channel on the traversed paths.

To deal with ISI, an OFDM transmitter inserts a cyclic prefix between consecutive symbols. The receiver discards the cyclic prefix and takes the remaining signal, thus eliminating any interference from adjacent symbols. This, however, does not work when we have a wideband interferer like the baby monitor. First, its signal may not have a cyclic prefix. Second, even if it does, as noted by past work on concurrent 802.11n transmissions [Tan et al. 2009], it is unlikely that the cyclic prefixes of the two devices are synchronized, in which case the receiver cannot discard a single cyclic prefix that eliminates ISI for both the devices.

The above discussion means that in the frequency domain, the interferer's signal, I_j, will experience ISI which would add noise. As a result, Equations 3.7 and 3.8 have additional noise terms due to ISI. While this is not a problem for the channel ratio estimation since one can average across more samples to obtain an accurate estimate of β_j; this additional noise would reduce the SNR for the signal of interest and, hence, affect its throughput.

The solution to the ISI problem is, however, simple. The 802.11n receiver needs to decode the signal of interest $s(t)$ by eliminating interference in the time domain. Here, ISI is simply a convolution with a filter, which can be removed by applying the inverse filter (i.e., an equalizer). Thus, we consider again the initial time domain Equations 3.5

and 3.6 which describe the signal at the 802.11n receiver:

$$y_1(t) = \mathbf{h}_i * i(t) + \mathbf{h}_s * s(t) \tag{3.13}$$

$$y_2(t) = \mathbf{h}'_i * i(t) + \mathbf{h}'_s * s(t). \tag{3.14}$$

We want to find a filter, **h**, such that:

$$\mathbf{h} * \mathbf{h}'_i = \mathbf{h}_i$$

Given such a filter, the receiver can convolve **h** with Equation 3.14 and subtract the resulting equation from Equation 3.13 to eliminate $i(t)$ and obtain an equation in $s(t)$, which it can decode using a standard 802.11 decoder.[10]

The above filter can be represented in the frequency domain as:

$$H_j H'_{ij} = H_{ij} \Rightarrow H_j = \frac{H_{ij}}{H'_{ij}} = \beta_j.$$

Thus, we can compute the desired filter **h** by taking the IFFT of the interferer channel ratios, β_j's, computed in Section 3.5.1.

To summarize, the 802.11n receiver first moves the received signal to the frequency domain where it computes the interferer channel ratios using Equation 3.12 while averaging over multiple samples to reduce the ISI and noise. Then, it transforms the interferer channel ratio into a time domain filter by taking an IFFT. Finally, it uses the filter to eliminate interference in the time domain. The receiver can now take this interference-free signal and decode its signal of interest using a standard 802.11 decoder.

3.5.3 Iterating to Increase Accuracy

The algorithm in Section 3.5.1 computes expectations by taking averages over multiple OFDM symbols. A packet, however, may not have enough OFDM symbols to obtain a highly accurate estimate. Also averaging over multiple packets will reduce TIMO's ability to deal with a dynamic interferer. Thus, in this section we are interested in obtaining an accurate estimate of the interferer's channel ratio, β_j, using only a few OFDM symbols.

To increase the accuracy of the estimate without much averaging, the receiver iterates over the following two steps:

Initialization. The receiver obtains a rough estimate of β_j by averaging over a limited number of OFDM symbols.

10. As described in Section 3.2, such a decoder would apply FFT and decode in the frequency domain.

Step 1. The receiver uses its estimate of β_j to obtain the signal, $s(t)$, as in Section 3.5.2. The receiver then decodes $s(t)$ using the standard decoder to obtain the transmitted bits.

Step 2. The receiver re-modulates the decoded bits to obtain an estimate of $s(t)$, which we call $\hat{s}(t)$. The receiver convolves $\hat{s}(t)$ with the channel functions and subtracts the results from $y_1(t)$ and $y_2(t)$. Thus, we obtain the following:

$$\hat{y}_1(t) = \mathbf{h}_i * i(t) + \mathbf{h}_s * (s(t) - \hat{s}(t))$$
$$\hat{y}_2(t) = \mathbf{h}'_i * i(t) + \mathbf{h}'_s * (s(t) - \hat{s}(t)).$$

The receiver then obtains a new estimate for β_j while treating $(s(t) - \hat{s}(t))$ as the new signal of interest.

After iterating between Steps 1 and 2 for two or three times, the receiver obtains an accurate estimate of the interferer's channel ratio β_j, which it uses to decode signal $s(t)$.

The reason why the above algorithm works is that in each iteration, the signal of interest used in Step 2, $(s(t) - \hat{s}(t))$, has a smaller magnitude. Since, in Step 2, the receiver is focused on estimating the interferer's ratio, the signal of interest plays the role of noise; reducing this signal's magnitude increases the accuracy of the ratio estimate. This higher accuracy in the ratio β_j percolates to the estimate of $s(t)$ in Step 1. Consequently, the decoded bits are more accurate and lead to even smaller difference between $\hat{s}(t)$ and $s(t)$, and hence an even more accurate β_j.

3.5.4 Estimating the 802.11n Channel Functions

So far, we have assumed that the 802.11n receiver knows the channel of the signal of interest, H_{sj} and H'_{sj}. To compute this channel we distinguish between two cases. First, the signal of interest starts before the interference in which case the receiver can use the 802.11 preamble to compute the channel, as usual. Second, the interference signal starts before the signal of interest. In this case, the receiver can easily compute the interferer's channel ratio $\beta_j = \frac{H_{ij}}{H'_{ij}}$ by taking the ratio of the signals it receives on its two antennas $Y_{1j} = H_{ij} I_j$ and $Y_{2j} = H'_{ij} I_j$. Once the receiver knows the interferer's channel ratio, it computes the equalization filter described in Section 3.5.2 and uses it to eliminate the interference signal. The receiver can then use the 802.11n preamble to compute the channel as usual.

Two points are worth noting. First, while it is easy to compute the interferer's channel ratios when the interferer is alone on the medium, this does not eliminate the need to continue tracking the interferer's channel ratio using the algorithm in Section 3.5.1. In particular, the channel ratio may change as the interferer moves to a different frequency, as in the narrowband phase of a microwave signal, or it might change for a mobile interferer, as with the cordless phone.

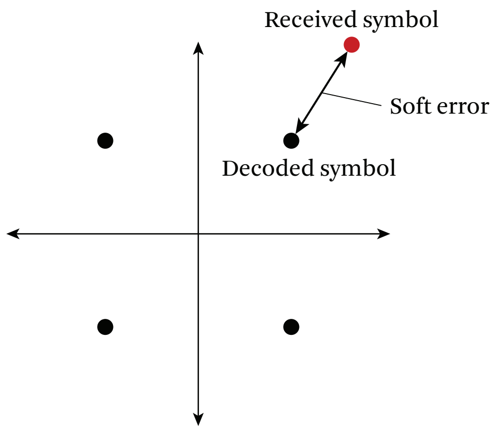
Second, the above scheme will miss in scenarios in which the interference and the 802.11n signal starts during the same OFDM symbol. This event has a low probability, and the resulting packet loss is minor in comparison to the packet loss observed without TIMO. When such an event occurs the packet will be retransmitted by its sender as usual.

3.5.5 Finding the Interference Boundaries

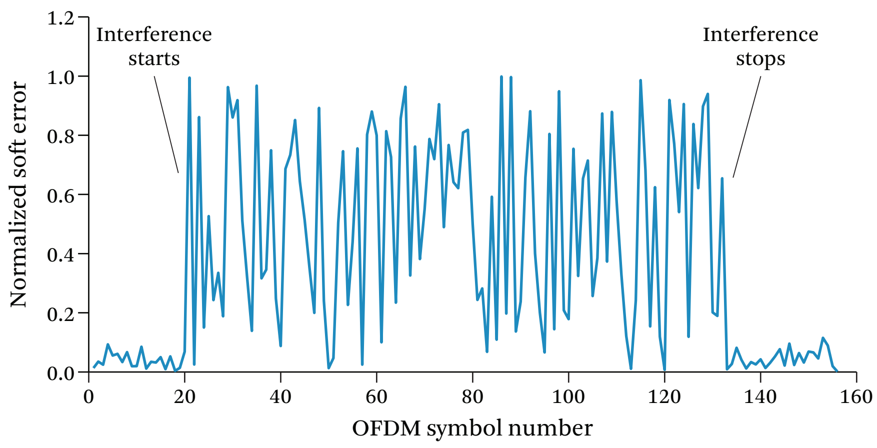
Estimating the interferer's channel ratio, β_j, using Equation 3.12 requires the 802.11n receiver to compute the expectations by taking averages over multiple OFDM symbols. This averaging, however, needs to be done only over symbols that are affected by interference. Thus, the 802.11n receiver needs to determine where, in a packet, interference starts and where it stops. The question of identifying the sequence of symbols affected by interference has been addressed in few recent systems, like PPR [Jamieson and Balakrishnan 2007] and SoftRate [Vutukuru et al. 2009]. Our approach follows the same principles. Specifically, when the interference signal starts, it causes a dramatic increase in decoding errors. As shown in Figure 3.6(a), these errors appear at the PHY layer as large differences between the received symbol and the nearest constellation points in the I and Q diagram. We refer to these differences as soft errors. Thus, for each OFDM subcarrier, the 802.11n receiver computes the soft error, and normalizes it by the minimum distance of the constellation. As shown in Figure 3.6(b), when the interferer starts, the soft errors jumps; when it ends, they go back to their low values. In our implementation we consider a jump that is higher than doubling the errors as a potential interferer, i.e., interference above 3 dB. This means that we might miss low power interferers, but such interferers can be dealt with using traditional methods like reducing the bit rate.

3.5.6 Putting it together

A TIMO receiver first performs packet detection as usual by looking for jumps in received power (using standard window detection algorithms [Heiskala and Terry 2001]). Then, the receiver computes the 802.11 preamble cross-correlation, in a manner similar to current 802.11. If the cross-correlation stays low, the receiver works under the assumption that the signal of interest may start later. Hence, it computes the channel ratios for the signal though it is not its signal of interest. On the other hand, if the cross-correlation spikes, the receiver identifies the packet as a signal of interest. It continues decoding the packet using a standard 802.11 decoder [IEEE 2012]. If the packet does



(a) Soft errors in a 4-QAM constellation



(b) Soft errors with interference

Figure 3.6 **Soft errors increase in the presence of interference.**

not pass the checksum test, the receiver computes the soft errors, as described in Section 3.5.5. If the soft errors jump by over 3 dB, the receiver initiates the channel ratio estimation algorithm. Specifically, for each OFDM bin, the TIMO decoder starts at the symbol where the soft errors jump and proceeds to compute the interference channel ratios in an iterative manner, as described in Section 3.5.3. Once the channel ratios are estimated for each OFDM subcarrier, the receiver uses the decoder in Section 3.5.2 to decode its signal of interest.

3.5.7 Complexity

While past work that deals with cross-technology interference[11] [Taher et al. 2008] typically employs different mechanisms for different technologies, TIMO is technology agnostic and hence its complexity stays constant as the number of technologies in the ISM band increases. Further, the components used in TIMO, such as correlation, equalization, and projection, are also used in MIMO receivers (although for a different purpose), and hence are amenable to hardware implementations.

3.6 Ensuring The Interferer Can Decode

A MIMO transmitter can also encode its signal to prevent interference to a competing transmission from a different technology. Specifically, let $i(t)$ be the competing signal and $s_1(t)$ and $s_2(t)$ the two streams that a two-antenna 802.11n node transmits. The receiver of the competing signal receives the following:

$$z(t) = h_i i(t) + h_{s1} s_1(t) + h_{s2} s_2(t),$$

where h_i refers to the channel from its transmitter and h_{s1} and h_{s2} are the channels from the two-antenna 802.11n transmitter. The 802.11n transmitter can cancel its signal at the receiver of the competing technology by ensuring that the signals it transmits on its two antennas satisfy $s_2(t) = -\frac{h_{s1}}{h_{s2}} s_1(t)$. Such a technique is referred to as interference nulling [Tse and Vishwanath 2005].[12]

We note that nulling does not require the knowledge of the exact channels to the receiver. It is sufficient to know the channel ratios to null the signal at some receiver. This is crucial since for cross-technology scenarios, it is hard to estimate the exact channel.

But how does the 802.11n transmitter compute the channel ratio to the interferer's receiver? If the interfering technology is bidirectional in the frequency of interest, then our 802.11n nodes can use the interference caused by the receiver's response to compute the channel ratio from the receiver to itself. This can be done by leveraging the algorithm in Section 3.5.1. The required ratio for nulling, however, refers to the channels in the opposite direction, i.e., from our 802.11n transmitter to the interfering receiver. To deal with this issue, TIMO exploits that wireless channels exhibit reciprocity, i.e., the channel function in the forward and backward direction is the same. Reciprocity

11. http://www.cisco.com/en/US/netsol/ns1070/index.html; last retrieved May 2014.

12. Note that having the 802.11n transmitter perform interference nulling does not require any modification to decoding at the 802.11n receiver.

is a known property that has been validated empirically by multiple studies [S. Gollakota et al. 2009, Zetterberg 2011].[13] Using reciprocity, one can compute the required channel ratio. Once the ratio is computed, the transmitter can perform interference nulling. We note that since it is hard to synchronize wideband cross-technology interferers with 802.11, to avoid ISI we perform nulling by using a time-domain equalizer similar to Section 3.5.2.

Thus, interference nulling combined with our algorithm for estimating the interferer's channel ratio provide a new primitive that enables a MIMO node to transmit in the presence of a different technology without hampering reception of that technology. This primitive, however, requires the competing technology to be bidirectional, i.e., the competing receiver acks the signal or transmits its own messages, like a cordless phone.

If the technology is bidirectional, then the MIMO transmitter can learn the channel ratio to the communicating node pair, using the interference they create. The MIMO transmitter then alternates between nulling its signal at the two communicating nodes. For example, in the case of a cordless phone, the 802.11 transmitter has to switch between nulling its signal at the handset and nulling its signal at the base. In the case of the cordless phone, the switching time is constant, and for the tested phone it is 2.25 ms. Even if the switching time is not constant, as long as the pattern of the interference is persistent (e.g., one data packet, followed by one ACK), the MIMO node can monitor the medium and immediately switch every time the medium goes idle.

On the other hand, if the receiver of the competing technology is not bidirectional, an 802.11n device has no way to compute its channel ratio, and hence cannot cancel its signal at the receiver of the competing technology. The impact of such interference will depend on the competing technology. For example, interference does not hamper a microwave oven function. Also, analog devices (e.g., an analog camera) have some level of resistance to interference which causes smooth degradation in their signal, and while they suffer from interference, they can still function if the interferer is not in close proximity (see Section 3.8).

In general, our objective is to create a form of coexistence between 802.11n and high-power interferers that approaches the coexistence it enjoys with low-power devices like Bluetooth, where the two technologies may interferer if they are in close proximity but the interference is limited and does not cause either device to become completely dys-

13. To use it in our system, one needs to calibrate the effect of the hardware before applying reciprocity. This calibration, however, is done once for the hardware. Furthermore, an 802.11n transmitter can perform this task without the help of any other node because it merely involves taking the difference between the two transmit chains attached to its two antennas.

functional. Unidirectional devices which do not sense the medium or use any feedback from their receiver tend to show some level of resistance to interference. Hence, even if the 802.11n node did not cancel its interference at their receiver, they can still support some level of coexistence, as long as 802.11n can protect itself from their interference.

3.7 Implementation

We have built a prototype of TIMO using the USRP2 radio platform and the GNURadio software package. A 2×2 MIMO system is built using two USRP2 radio-boards connected via an external clock.[14] Each USRP2 is configured to span a 10 MHz channel by setting both the interpolation rate and decimation rate to 10. The resulting MIMO node runs a PHY layer similar to that of 802.11n, i.e., it has 64 OFDM subcarriers, a modulation choice of BPSK, 4-QAM, 16-QAM, or 64-QAM, and punctured convolution codes with standard 802.11 code rates [IEEE 2012]. Since we operate at half the 802.11 bandwidth, the possible bit rates span 3 to 27 Mbps. We modify the receiver MIMO decoding algorithm to incorporate TIMO (summarized in Section 3.5.6). We also implemented interference nulling at the MIMO transmitters. To work with cross-technology interference, the transmitter first computes the channel ratios and then uses them for nulling (as described in Section 3.6).

3.8 Performance Evaluation

We evaluate TIMO with three high-power interferers: a DSSS cordless phone, a microwave oven, and a baby monitor.

3.8.1 Cordless Phone

Again, we use the Uniden TRU 4465-2 cordless phone as the interferer. We also use the same testbed in Figure 3.1.

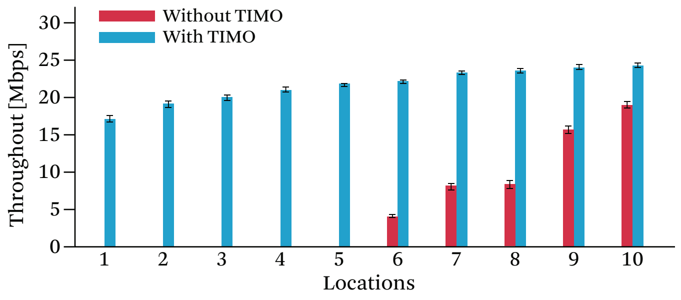
Addressing Cross-Technology Interference. We first evaluate TIMO's ability to help 802.11n nodes operate in the presence of high power cross-technology interference. We place two USRP-based 802.11n nodes in locations A and B in Figure 3.1. In each run, we place the cordless phone system in one of the 10 interferer locations in Figure 3.1. We transfer a 20 MB file between the 802.11n pair at the best bitrate for the channel in the presence of interference from the cordless phone. This rate is determined by initially trying all the possible bitrates and choosing the one which yields

14. http://www.jackson-labs.com; last retrieved May 2014.

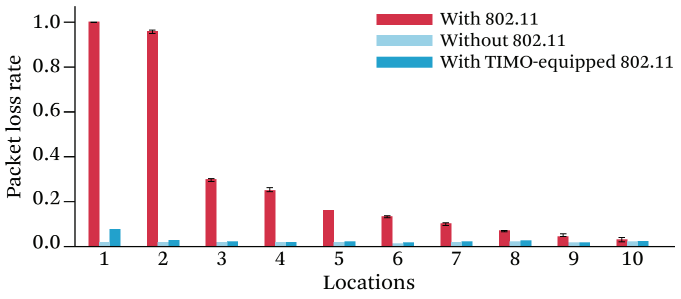
the highest throughput for the rest of the run. The 802.11 receiver logs the received samples and processes them both with and without TIMO.

Note that in contrast to the experiments done with commercial 802.11n nodes, the USRP implementation of 802.11n does not use carrier sense. Carrier sense is hard to implement in software due to its strict timing requirements. This constraint, however, can be beneficial. In particular, the lack of carrier sense provides insight into whether the throughput loss of commercial 802.11n is due to the nodes sensing the phone's signal and abstaining from transmitting, or due to their packets being corrupted by interference.

Figure 3.7(a) plots the throughput of the 802.11 MIMO nodes in the presence of the phone signal, with and without TIMO. The figure reveals the following.



(a) 802.11 throughput with and without TIMO in the presence of interference from DSSS phone



(b) Packet loss at the DSSS phone with and without TIMO

Figure 3.7 **Interference from a DSSS cordless phone.** (a) shows that TIMO significantly improves the throughput of 802.11 USRP2-based nodes in the presence of interference from a DSSS phone. (b) shows that if 802.11 nodes transmit concurrently with a DSSS cordless phone, they can cause the phone a dramatic packet loss at close distances. TIMO, however, enables such nodes to transmit concurrently with the phone without hampering its performance.

- Without TIMO, interference from the cordless phone causes the 802.11 nodes to completely lose connectivity in half of the testbed locations. This loss of connectivity occurs even though the nodes have deactivated carrier sense and are using the best bit rate for the channel. This means that the interference in these locations is too high even for the lowest bit rate supported by 802.11. This loss in connectivity can be attributed to the fact that the phone system transmits continuously at a high power. Hence, the 802.11 packets are always subject to strong interference. As the interferer moves away from the 802.11 USRP-based nodes, their throughput improves because of reduced interference.
- In contrast, with TIMO, the 802.11 nodes never experience disconnectivity. Also, their throughput becomes much higher and close to optimal (24.5 Mbps) at most locations. The throughput decreases slightly as the phone moves closer to the 802.11 receiver in location B because of residual interference, but continues to be 78% of the optimal throughput even when the phone is 1 foot away from the 802.11 receiver. These results indicate that TIMO is successful at exploiting MIMO capability to address 802.11 cross-technology interference.
- Comparing the throughput of the USRP-based 802.11n implementation to that of commercial 802.11n in Section 3.1 shows that while carrier sense contributed to the loss of connectivity particularly when the interferers are in locations 6–10, it is not the main reason since even though the USRP nodes do not implement carrier sense, they still lose connectivity in 50% of the locations.

Transmitting without Harming the Competing Technology. Next, we evaluate TIMO's ability to allow 802.11n to transmit concurrently with a cordless phone in the same frequency band, but without harming the phone's transmission. The commercial phone does not give us access to packets, making it hard to evaluate the impact of TIMO's interference nulling. Instead, we implement the phone's physical layer in GNURadio and experiment with a USRP-based DSSS phone. We try to match the physical layer description of the Uniden phone. In particular, the transmitter feeds digital bits to a scrambler, differential encoder, and a spread spectrum module. The spread spectrum module sends bits at a data rate of 1.366 Mbps over FSK modulation. The receiver computes the correlation with the spreading code and outputs the data bits. For every packet we use the CRC to detect if it was correctly received.

We place the USRP nodes that perform the role of the phone base and handset at location A and B in the testbed. We then place a 802.11 USRP transmitter at each of locations 1–10 in the testbed, and let it transmit at the same time as the USRP phone. The 802.11 USRP transmitter uses TIMO to null its signal at the phone.

The 802.11 transmitter has to alternate between nulling its signal at the phone base and the handset. Since the Uniden phone packets have a fixed duration of 2.25 ms[15] this switching can easily happen on 802.11 hardware. However, due to the software nature of GNURadio, it is hard to alternate with the phone system at a granularity of about 2.25 ms. Thus, in our experiments, we increase the inter-packet time and the packet duration to 20 ms, which allows us to alternate with the phone system in software.

Each run of the experiment has three parts. First, the phone handset and base exchange packets without any interference from the 802.11n transmitter. Next, the handset and base exchange packets with interference from the 802.11 node but without TIMO. Finally, the handset and base exchange packets concurrently with the 802.11n node which uses TIMO.

Figure 3.7(b) shows the packet loss rate at the handset for the above three cases. The figure shows three main trends.

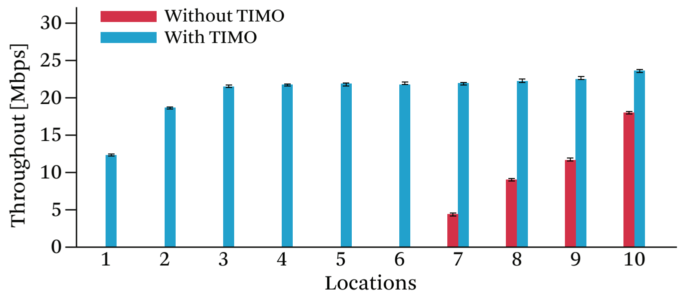
- In comparison with 802.11n, the DSSS phone is more resilient to cross-technology interference. This is due to its use of FSK combined with a high redundancy DSSS code. Despite this resilience, without TIMO, the phone suffers a high loss rate at locations close to the 802.11 nodes.
- In contrast, TIMO significantly reduces the loss rate at the handset across all the locations. Further, in locations 2–10 the loss rate is almost as low as that without any interference. We note that this is true even for locations where the interferer is closer to the handset than the base is to the handset (locations 2–4). Thus, we conclude that TIMO can help 802.11 and DSSS phones coexist.
- Finally, when the 802.11 interferer is less than a foot from the handset (location 1), the packet loss rate is higher than that without interference. This is because, in practice, it is difficult to completely eliminate interference using interference nulling. The residual interference may cause an increase in packet loss rate at such close distances. However, even at location 1, while TIMO did not completely eliminate interference, it still reduces the packet losses by more than 14×, from 100% to about 6–7%.

3.8.2 Baby Monitor

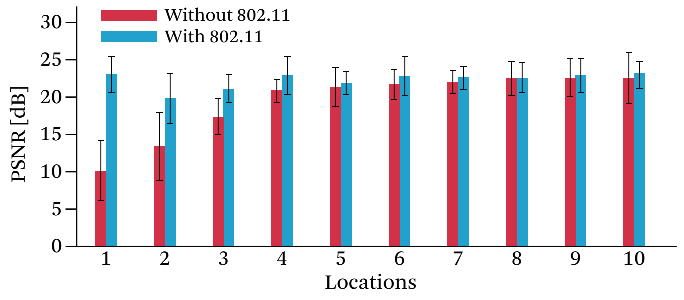
Next, we evaluate TIMO with a baby monitor.

Impact of baby monitors on 802.11n. To evaluate this, we repeat the previous experiment after replacing the microwave with the C-501 baby monitor. For every interferer location, we run the system with and without TIMO, and plot the results in Figure 3.8(a).

15. http://www.uniden.com; last retrieved May 2014.



(a) 802.11 throughput with and without TIMO in the presence of interference from a baby monitor



(b) Camera PNSR (above 20 dB is watchable; above 25 is good [156])

Figure 3.8 **Interference from a baby monitor.** (a) shows that TIMO significantly improves the throughput of 802.11 nodes in the presence of interference from a baby monitor. (b) shows that while TIMO cannot cancel its signal at the camera's receiver because it use a unidirectional communication, the camera's signal is watchable in all locations but the two closest to the 802.11 nodes.

The figure shows that TIMO significantly increases the throughput in the presence of interference from the tested baby monitor. In particular, without TIMO the 802.11 nodes experience complete disconnectivity for 60% of locations of the baby monitor. In contrast, with TIMO no scenario causes disconnectivity and the overall throughput is significantly higher. We note that in comparison to the performance of commercial 802.11n nodes, the USRP-based 802.11n implementation does not use carrier sense, and hence was able to transmit and obtain some throughput in scenarios where the commercial 802.11n nodes refrained from transmitting due to carrier sense.

Impact of 802.11n transmissions on baby monitors. Communication in the baby monitor system is one-way. The camera continuously broadcasts the analog video. A monitor in range of the device receives the signal, decodes it, and displays it on its screen. Given no signal from the video receiver, TIMO is limited in its ability to protect the transmitted video. Thus, we would like to check how the camera is affected by interference from our 802.11 implementation (which use the same power level as a laptop, i.e., about 30 mW).

To do so, we place the camera and its video receiver in locations A and B in the testbed. We move the 802.11-USRP node across the various interferer locations, and at each location, we ensure it interferes with the camera's transmission. We compare the received video quality with and without interference from 802.11. We measure video quality using PSNR, which is a standard video metric. A PSNR of less than 20 dB is hard to watch, whereas PSNRs in the range 25–30 dB are good. The PSNR can be computed only with respect to the original video. However, the camera does not provide us access to the original video before transmission over the wireless medium. To obtain a video baseline, we focus the camera on a static image for all experiments, and make it transmit the same frame 1000 times. Then, we take the average pixel value in these 1000 versions of the same frame and consider this to be the ground truth. All experiments are run with the camera focused on the same picture so that they can be compared with this ground truth.

Figure 3.8(b) shows the PSNR of the received video both with and without interference from our USRP-based 802.11 implementation. The figure shows that at the closest two locations, which are less than 6 feet away from the 802.11 interferer, the video is not watchable. However, for the rest of the locations, the video quality stays watchable. Further, for seven out of the ten testbed locations, the video PSNR hardly changes from its value without interference. This is expected because devices that blast without checking for interference or without feedback tend to be relatively resilient to some level of interference.

We note that since the monitoring system is unidirectional, TIMO cannot cancel its signal at potential video receivers; hence, we observe that interference degrades the monitoring system's performance at nearby locations. However, in contrast to the current mode of operation, where 802.11 loses connectivity in most locations due to interference, TIMO is an improvement over the status quo because it reduces the range of interference to close-by locations. This moves the system to a scenario where the two technologies enjoy some level of coexistence, which despite being far from optimal, is more acceptable than the current situation.

3.8.3 Microwave Oven

We evaluate TIMO's performance in the presence of interference from the microwave oven used in the experiments in Section 3.1. We repeat the experiment we conducted with the cordless phone, where we place the USRP-based 802.11 devices in locations

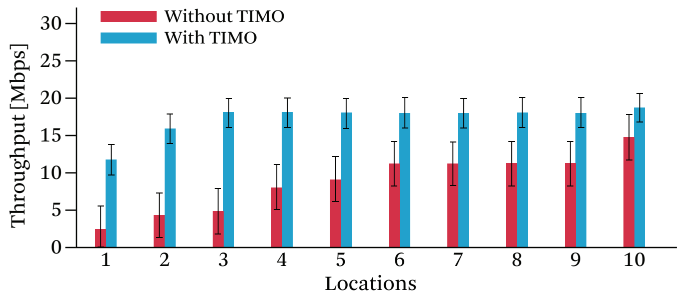


Figure 3.9 **802.11 throughput with interference from a microwave oven.** The figure shows that TIMO increases resilience to microwave interference.

A and B, and let them exchange traffic with the microwave on and off. We perform the experiment for each of the 10 interferer locations in the testbed. In each run, the 802.11 transmitter uses the best bitrate as in Section 3.8.1.

Figure 3.9 shows the average throughput and standard deviation, with and without TIMO. Without TIMO, the performance of the USRP2 nodes is relatively similar to that of the commercial 802.11n nodes. Specifically, at short distances, the throughput is very low due to increased interference. As the microwave is moved away, the nodes start getting packet through during the OFF periods of the microwave. In contrast, TIMO significantly increases resilience to interference from the microwave, allowing the 802.11 USRP node to deliver packets efficiently even during the ON periods of the microwave. Microwave ovens leak significantly high power during the ON periods, which could reach 1 Watt [Bandspeed 2010]. The results show that TIMO is effective even with such high-power interferers.

TIMO's approach is based on treating cross technology interference as if it were a stream from a single-antenna node of the same technology. Residential microwave ovens are equipped with a cavity magnetron which radiates energy in the 2.4 GHz range. Since they have only one magnetron radiating energy, theory concludes that they act as a single antenna device [Taher et al. 2008]. Our results confirm theoretical conclusions and show that TIMO can successfully treat a microwave as a single-antenna interferer.

3.8.4 Multiple Interferers

This experiment includes three node pairs with different transmission technologies: our 2×2 802.11n implementation, our DSSS phone implementation, and a GNURadio ZigBee implementation. The 802.11n devices occupy a 10 MHz channel, the DSSS phone occupies a 4 MHz channel, and the ZigBee devices occupy 5 MHz. The center frequencies of these devices are picked such that the phone interferes with the first half of the 802.11 channel, whereas the ZigBee device interferes with the second half. We place these six nodes randomly at the marked locations in Figure 3.1. We make the three pairs transmit concurrently, and we repeat each run with and without TIMO. As before, we make the inter-packet arrival and the packet duration for the cordless phone and ZigBee nodes 20 ms, to allow for a software implementation.

Figure 3.10(a) plots the CDF of 802.11 throughput with and without TIMO. The figure shows that without TIMO, about 67% of the locations cannot get any packets through and the average throughput is low. In contrast, with TIMO no locations suffer disconnectivity and the average throughput increases significantly.

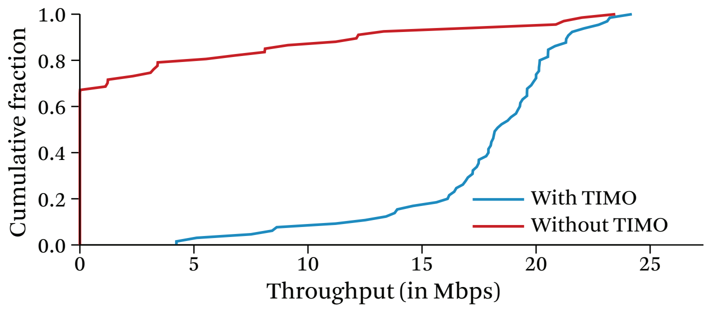
Figure 3.10(b) and 3.10(c) plot the packet loss rate of the competing technologies: the DSSS phone and ZigBee. The figure shows that if 802.11n transmits concurrently, without TIMO, these technologies can suffer significant packet loss. However, if 802.11n employs TIMO, then its interference increases loss rates by less than 0.5%, which is negligible. Thus, TIMO can help diverse technologies co-exist in the same frequency band while placing the burden of interference prevention on high-end MIMO nodes instead of low-end single antenna systems.

3.9 Micro Benchmarks

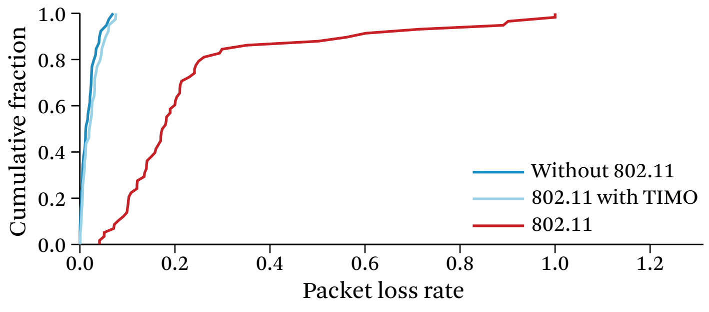
Finally, we zoom in on the components of TIMO to examine the tradeoff between averaging over a larger number of symbols and applying the same algorithm iteratively over a smaller number of symbols.

We transfer a 20 MB file between two 2×2 802.11 USRP2 nodes. A third USRP2 node plays the role of an unknown interfering technology, and transmits a signal unknown to the 802.11 USRP2 nodes. We run the experiment for random placement of the three nodes in various locations in Figure 3.1. We want to compute the amount of averaging and the number of iterations that TIMO needs to obtain an accurate estimate of the interferer's channel ratio. To obtain a ground truth of the channel ratios, we provide a baseline receiver with the full knowledge of the transmitted interference signal so that it can use the whole signal as if it were a preamble, and compute a very accurate estimate of the interferer's channel. We compute this estimate over periods of 1 ms each, which is significantly lower than the coherence time for indoor static channels at 2.4 GHz. For each run, we process the signal using the baseline receiver and TIMO.

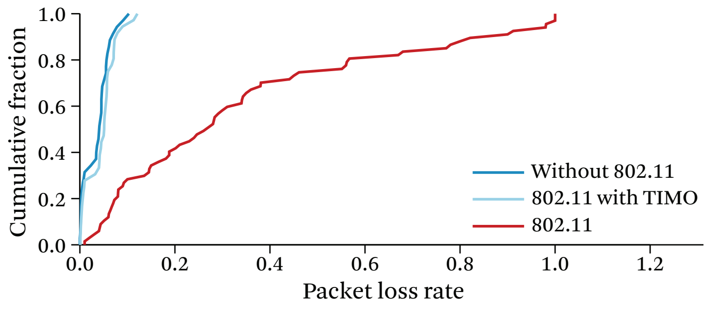
Figure 3.11 plots the number of symbols that TIMO needs to average over to obtain an estimate of the channel ratio that is within 3% of the value obtained with the baseline. The figure shows the results for the four modulations in 802.11 (BPSK, 4-QAM, 16-QAM, and 64-QAM). The plots reveal the following trends.



(a) 802.11 throughput



(b) ZigBee throughput



(c) DSSS phone throughput

Figure 3.10 **TIMO with multiple interferers.** The figure shows the throughput CDFs for three technologies that are transmitting concurrently in overlapping frequencies: 802.11n, DSSS phone, and ZigBee.

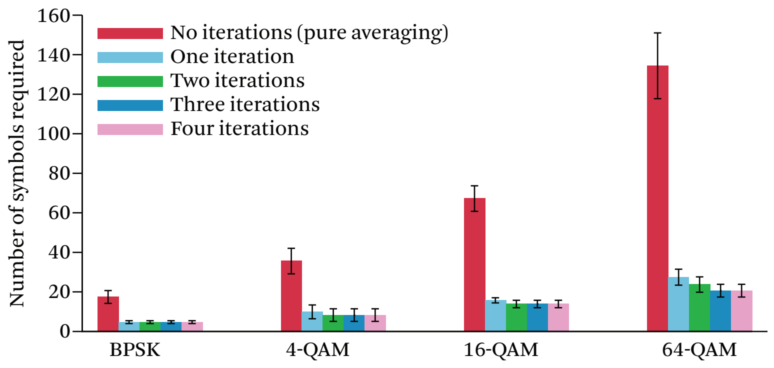


Figure 3.11 **Tradeoff between the number of averaged symbols and the number of iterations.** With three iterations, TIMO can achieve the same accuracy as a baseline that knows the structure and the preamble of the interferer, while maintaining the averaged symbols less than 22 for all modulations.

- The iterative algorithm yields a significant reduction in the number of symbols required to average over to obtain an accurate estimate of the interferer's channel ratio.
- Across all modulation schemes, two to three iterations are sufficient, and the return from more iterations is negligible. The reason why there is a ceiling for the iteration gain is that iterating does not provide more information; it only provides a better estimation using the collected information. After some point, the algorithm becomes limited by the intrinsic noise in the collected measurements.
- Given three iterations, TIMO needs to average over less than 22 symbols even at the highest modulation scheme.

3.10 Related Work

Wireless interference has been the topic of much recent research. Work in this area falls under two broad categories.

(a) Interference Across Technologies. One can identify three main approaches within this category. The first approach attempts to eliminate interference by isolating the signals in time, frequency, or space. The most common isolation approach is to employ frequency-based isolation, such as OFDM subcarrier suppression [Mishra et al. 2007, Rahul et al. 2008, He et al. 2010], variable channel width [Chandra et al. 2008], or other fine grained frequency fragmentation techniques [Yang et al. 2010, Cao et al.

2010, Moscibroda et al. 2008]. TIMO, on the other hand, enables independent technologies to share the same frequencies without interfering with each other. Directional antennas may also be used to provide spatial isolation and reduce interference. However, directional antennas are difficult to use in indoor scenarios where the signal tends to bounce off walls and furniture and scatter around [Tse and Vishwanath 2005]. In contrast, TIMO works in scattering environments and applies even when the two receivers are in the same direction.

The second approach uses mitigation schemes to modify transmissions to be more resilient to interference (e.g. by coding or by lowering the bit rates). Mitigation proposals like PPR [Jamieson and Balakrishnan 2007], though designed and evaluated for the same technology, can work across technologies. These schemes however assume interference is fairly transient and limited to some bytes in each packet. In contrast, TIMO can deal with persistent interference.

Finally, some proposals identify the type of interference (is it ZigBee? Bluetooth?) and inform the user so he may switch off the interfering device[16] [Lakshminarayanan et al. 2009]. Others leverage the specific characteristic of a particular technology to design a suitable coexistence strategy [Taher et al. 2008]. Like this prior work, TIMO aims to provide coexistence of different wireless technologies. TIMO provides a single approach that works with different technologies, e.g., microwave ovens, cordless phones, etc, and applies even to unknown technologies.

(b) Interference from the Same Technology. Recent work in this category include interference cancellation [Halperin et al. 2007], ZigZag [Gollakota and Katabi 2008], and analog network coding [Katti et al. 2007] which address the problem of interference from other 802.11 nodes. The closest to ours is prior work on MIMO systems which enables multiple transmitters to transmit concurrently without interference. This includes schemes like SAM [Tan et al. 2009], Interference Alignment and Cancellation [S. Gollakota et al. 2009], and beamforming systems [Aryafar et al. 2010]. Unlike these schemes, however, TIMO delivers a MIMO system that enables cooperation with multiple different wireless technologies.

Finally, TIMO is related to prior work on interference management in cellular networks, which uses multiple antennas to mitigate interference from nodes operating in adjacent cells [Tse and Vishwanath 2005].[17] In contrast to this work, however, TIMO develops new algorithms that can address cross-technology interference.

16. http://www.cisco.com/en/US/netsol/ns1070/index.html; last retrieved May 2014.

17. http://www.arraycomm.com; last retrieved May 2014.

3.11 Discussion

Finally, we would like to comment on the scope of TIMO. TIMO presents a MIMO design that enables 802.11n to communicate in the presence of high-power cross-technology interference. TIMO exploits 802.11n's MIMO capability to treat a high-power signal from a different technology as if it were another stream from the same technology, hence enabling diverse technologies to share the same frequency band. We show via a proof-of-concept implementation that TIMO enables 802.11n to communicate effectively in the presence of typical interferers.

TIMO also provides a basic framework for a new form of coexistence, in which different technologies do not necessarily have to find unoccupied bands and could, in crowded environments, occupy the same band, thus increasing spectral efficiency. Further, beyond 802.11n, the algorithms and techniques presented in this work generalize to other technologies and frequency bands. Specifically, cross-technology interference is a general problem faced not only by 802.11n, but also technologies in the new white space frequencies [Bahl et al. 2009] that suffer from cross-technology interference, including TV broadcasts and wireless microphone transmissions. TIMO's approach of using occupied frequencies can, in principle, be used with these technologies to enable better coexistence and higher spectral utilization.

4

Non-Invasive Approach to Securing Medical Implants

The past few years have produced innovative health-oriented networking and wireless communication technologies, ranging from low-power medical radios that harvest body energy [Koplow et al. 2008] to wireless sensor networks for in-home monitoring and diagnosis [Shnayder et al. 2005]. Today, such wireless systems have become an intrinsic part of many modern medical devices [Panescu 2008]. In particular, implantable medical devices (IMDs), including pacemakers, cardiac defibrillators, insulin pumps, and neurostimulators, all feature wireless communication [Panescu 2008]. Adding wireless connectivity to IMDs has enabled remote monitoring of patients' vital signs and improved care providers' ability to deliver timely treatment, leading to a better health care system [Maisel 2005].

Recent work, however, has shown that such wireless connectivity can be exploited to compromise the confidentiality of the IMD's transmitted data or to send the IMD unauthorized commands—even commands that cause the IMD to deliver an electric shock to the patient [Halperin et al. 2008b, Halperin et al. 2008c]. In other systems, designers use cryptographic methods to provide confidentiality and prevent unauthorized access. However, adding cryptography *directly* to IMDs themselves is difficult for the following reasons.

Inalterability. In the U.S. alone, there are millions of people who already have wireless IMDs, and about 300,000 such IMDs are implanted every year [Zhan et al. 2007]. Once implanted, an IMD can last up to 10 years [Fu 2009], and replacing it requires surgery that carries risks of major complications. Incorporating cryptographic mechanims into existing IMDs may be infeasible because of limited device memory and hence can only be achieved by replacing the IMDs. This is not an option for people who have IMDs or may acquire them in the near future.

Safety. It is crucial to ensure that health care professionals always have immediate access to an implanted device. However, if cryptographic methods are embedded in the IMD itself, the device may deny a health care provider access unless she has the right credentials. Yet, credentials might not be available in scenarios where the patient is at a different hospital, the patient is unconscious, or the cryptographic key storage is damaged or unreachable [Halperin et al. 2008c, Maisel 2005]. Inability to temporarily adjust or disable an IMD could prove fatal in emergency situations.[1]

Maintainability. Software bugs are particularly problematic for IMDs because they can lead to device recalls. In the last eight years, about 1.5 million software-based medical devices were recalled [Fu 2011]. Between 1999 and 2005, the number of recalls of software-based medical devices more than doubled; more than 11% of all medical-device recalls during this time period were attributed to software failures [Fu 2011]. Such recalls are costly and could require surgery if the model is already implanted. Thus, it is desirable to limit IMDs' software to only medically necessary functions.

This work explores the feasibility of protecting IMDs *without modifying them* by implementing security mechanisms entirely on an external device. Such an approach enhances the security of IMDs for patients who already have them, empowers medical personnel to access a protected IMD by removing the external device or powering it off, and does not in itself increase the risk of IMD recalls.

We present a design in which an external device, called the *shield*, is interposed between the IMD and potential counter-parties, e.g., worn on the body near an implanted device. The shield acts as a gateway that relays messages between the IMD and authorized nodes. It uses a novel physical-layer mechanism to secure its communication with the IMD, and it uses a standard cryptographic channel to communicate with other authorized nodes.

The shield counters two classes of adversarial actions: passive eavesdropping that threatens the confidentiality of the IMD's transmissions, and active transmission of unauthorized radio commands to the IMD. First, to provide confidentiality for the IMD's transmissions, the shield continuously listens for those transmissions and jams them so that they cannot be decoded by eavesdroppers. The shield uses a novel radio design to simultaneously receive the IMD's signal and transmit a jamming signal. The shield then transmits the IMD's signal to an authorized endpoint using standard

1. Note that distributing the credentials widely beyond the patient's primary health care providers increases the probability of the key being leaked and presents a major key revocation problem.

cryptographic techniques. Second, to protect the IMD against commands from unauthorized endpoints, the shield listens for unauthorized transmissions addressing the IMD and jams them. As a result of jamming, the IMD cannot decode the adversarial transmissions, and hence the adversary fails to make the IMD execute an unauthorized command.

A key challenge that we had to overcome to realize this architecture is to design a small wearable radio that simultaneously jams the IMD's signal and receives it. We build on prior work in the area of full-duplex radio design, which enables a single node to transmit and receive simultaneously [Choi et al. 2010, Duarte and Sabharwal 2010]. However, prior work requires large antenna separation and hence yields large devices unsuitable for our application. In particular, the state-of-the-art design for full-duplex radios [Choi et al. 2010] exploits the property that a signal reverses its phase every half a wavelength; it transmits the same signal from two antennas and puts a receive antenna *exactly* half a wavelength closer to one of the transmit antennas than the other. An antenna separation of half a wavelength, however, is unsuitable for our context: the IMDs we consider operate in the 400 MHz band [Federal Communications Commission 1999] with a wavelength of about 75 cm. A shield that requires the antennas to be rigidly separated by exactly half a wavelength (37.5 cm) challenges the notion of wearability and therefore patient acceptability.

This work presents a full-duplex radio that does not impose restrictions on antenna separation or positioning, and hence can be built as a small wearable device. Our design uses two antennas: a jamming antenna and a receive antenna, placed next to each other. The jamming antenna transmits a random signal to prevent eavesdroppers from decoding the IMD's transmissions. However, instead of relying on a particular positioning to cancel the jamming signal at the receive antenna, we connect the receive antenna simultaneously to both a transmit and a receive chain. We then make the transmit chain send an *antidote* signal that cancels the jamming signal at the receive antenna's front end, allowing it to receive the IMD's signal and decode it. We show both analytically and empirically that our design delivers its security goals without antenna separation; hence, it can be built as a small wearable radio.

Our design has additional desirable features. Specifically, because the shield can receive while jamming, it can detect adversaries who try to alter the shield's signal to convey unauthorized messages to the IMD. It can also ensure that it stops jamming the medium when an adversarial signal ends, allowing legitimate devices to communicate.

We have implemented a prototype of our design on USRP2 software radios. We use 400 MHz daughterboards for compatibility with the 402–405 MHz Medical Implant Communication Services (MICS) band used by IMDs [Federal Communications Commission 1999]. We evaluate our prototype shield against two modern IMDs, namely, the

Medtronic Virtuoso implantable cardiac defibrillator (ICD) and the Concerto cardiac resynchronization therapy device (CRT).[2] Our evaluation reveals the following.

- When the shield is present, it jams the IMD's messages, causing even nearby (20 cm away) eavesdroppers to experience a bit error rate of nearly 50%, which is no better than a random guess.
- When the shield jams the IMD's packets, it can still reliably decode them (the packet loss rate is 0.2%, which is negligible). We conclude that the shield and the IMD share an information channel that is inaccessible to other parties.
- When the shield is absent, the IMD replies to unauthorized commands, even if the adversary is in a non-line-of-sight location more than 14 m away, and uses a commercial device that operates in the MICS band and adheres to the FCC power limit.
- When the shield is present and has the same transmit power as the adversary, the IMD does not respond to unauthorized commands, even when the adversary is only 20 cm away.
- When the shield is absent and an adversary with 100 times the shield's power transmits unauthorized commands, the IMD responds from distances as large as 27 m. When the shield is present, however, the high-powered adversary's attempts succeed only from distances less than 5 m, and only in line-of-sight locations. The shield always detects high-powered adversarial transmissions and raises an alarm. We conclude that sufficiently high-powered adversaries present an intrinsic limitation to our physical-layer mechanism. However, the shield's presence reduces the adversary's success range and informs the patient, raising the bar for the adversary's attempts.

The shield is, to our knowledge, the first system that simultaneously provides confidentiality for IMDs' transmissions and protects IMDs against commands from unauthorized parties *without requiring any modification to the IMDs themselves*. Further, because it affords physical-layer protection, it may also help provide a complementary defense-in-depth solution to devices that feature cryptographic or other application-layer mechanisms.

Disclaimer. Operating a jamming device has legal implications that vary by jurisdiction and frequency band. The definition of jamming also depends on both context and

2. http://www.medtronic.com/; last retrieved May 2014.

intent. Our experiments were conducted in tightly controlled environments where no patients were present. Further, the intent of a shield is never to interfere with communications that do not involve its protected IMD. We recommend that anyone considering deployment of technology based on this research consult with their own legal counsel.

4.1 IMD Communication Primer

Wireless communication appears in a wide range of IMDs, including those that treat heart failure, diabetes, and Parkinson's disease. Older models communicated in the 175 KHz band [Halperin et al. 2008c]. However, in 1999, the FCC set aside the 402–405 MHz band for medical implant communication services (MICS) [Federal Communications Commission 1999]. The MICS band was considered well suited for IMDs because of its international availability for this purpose [European Telecommunications Standard Institute 2009], its signal propagation characteristics in the human body, and its range of several meters, which allows remote monitoring. Modern IMDs communicate medical information in the MICS band, though devices may use other bands for activation (e.g., 2.4 GHz or 175 KHz) [Sagan 2007]. IMDs share the MICS band with meteorological systems on a secondary basis and should ensure that their usage of it does not interfere with these systems. The FCC divides the MICS band into multiple channels of 300 KHz width [Federal Communications Commission 1999]. A pair of communicating devices uses one of these channels.

IMDs typically communicate infrequently with a device called an IMD programmer (hereafter, *programmer*). The programmer initiates a session with the IMD during which it either queries the IMD for its data (e.g., patient name, ECG signal) or sends it commands (e.g., a treatment modification). By FCC requirement, the IMD does not normally initiate communications; it transmits *only* in response to a transmission from a programmer [Federal Communications Commission 1999] or if it detects a life-threatening condition [Industry Canada 2010].

A programmer and an IMD share the medium with other devices as follows [Federal Communications Commission 1999]. Before they can use a 300 KHz channel for their session, they must "listen" for a minimum of 10 ms to ensure that the channel is unoccupied. Once they find an unoccupied channel, they establish a session and alternate between the programmer transmitting a query or command and the IMD responding immediately without sensing the medium [International Telecommunications Union 1998]. The programmer and IMD can keep using the channel until the end of their session, or until they encounter persistent interference, in which case they listen again to find an unoccupied channel.

4.2 Assumptions and Threat Model

4.2.1 Assumptions

We assume that IMDs and authorized programmers are honest and follow the protocols specified by the FCC and their manufacturers. We also assume the availability of a secure channel for transmissions between authorized programmers and the shield; this channel may use the MICS band or other bands. We further assume that the shield is a wearable device located close to the IMD, such as a necklace. Wearable medical devices are common in the medical industry [Ostrovsky 2009, Scheffler et al. 2003]. We also assume that the adversary does not physically try to remove the shield or damage it. We assume that legitimate messages sent to an IMD have a checksum and that the IMD will discard any message that fails the checksum test. This latter assumption is satisfied by all wireless protocols that we are aware of, including the ones used by the IMDs we tested (Section 4.8). Finally, we assume that the IMD does not normally initiate transmissions (in accordance with FCC rules [Federal Communications Commission 1999]); if the IMD initiates a transmission because it detects a life-threatening condition, we make no attempt to protect the confidentiality of that transmission.

4.2.2 Threat Model

We address two classes of commonly considered radio-equipped adversaries: passive eavesdroppers that threaten the confidentiality of the IMD's transmissions, and active adversaries that attempt to send unauthorized radio commands to the IMD [Fu 2011, Maisel and Kohno 2010].

(a) Passive Eavesdropper. Such an adversary eavesdrops on the wireless medium and listens for an IMD's transmissions. We allow this adversary the following properties.

- The adversary may try different decoding strategies. It may consider the jamming signal as noise and try to decode in the presence of jamming. Alternatively, it can implement interference cancellation or joint decoding in an attempt to simultaneously decode the jamming signal and the IMD's transmission. However, basic results in multi-user information theory show that decoding multiple signals is impossible if the total information rate is outside the capacity region [Tse and Vishwanath 2005]. We ensure that the information rate at the eavesdropper exceeds the capacity region by making the shield jam at an excessively high rate; the jamming signal is random and sent without modulation or coding.

- For the purpose of this book, we assume that the adversary can have as many antennas as the shield. In [Gollakota et al. 2011], we evaluate how our system works in the presence of an adversary with more antennas than the shield.
- The adversary may be in any location farther away from the IMD than the shield (e.g., at distances 20 cm and greater).

(b) Active adversary. Such an adversary sends unauthorized radio commands to the IMD. These commands may modify the IMD's configuration or trigger the IMD to transmit unnecessarily, depleting its battery. We allow this adversary the following properties.

- The adversary may use one of the following approaches to send commands: it may generate its own unauthorized messages; it may record prior messages from other sources and play them back to the IMD; or it may try to alter an authorized message on the channel; for example, by transmitting at a higher power and causing a capture effect at the IMD [Santhapuri et al. 2008].
- The adversary may use different types of hardware. The adversary may transmit with a commercial IMD programmer acquired from a hospital or elsewhere. Such an approach does not require the adversary to know the technical specifications of the IMD's communication or to reverse-engineer its protocol. However, an adversary that simply uses an unmodified commercial IMD programmer cannot use a transmit power higher than that allowed by the FCC. Alternatively, a more sophisticated adversary might reverse-engineer the IMD's communication protocol, then modify the IMD programmer's hardware or use his own radio transmitter to send commands. In this case, the adversary can customize the hardware to transmit at a higher power than the FCC allows.
- The adversary may be in any location farther away from the IMD than the shield.

4.3 System Overview

To achieve our design goal of protecting an IMD without modifying it, we design a device called the *shield* that sits near the IMD and acts as a proxy. An authorized programmer that wants to communicate with the IMD instead exchanges its messages with the shield, which relays them to the IMD and sends back the IMD's responses, as shown in Figure 4.1. We assume the existence of an authenticated, encrypted channel between the shield and the programmer. This channel can be established using TEP from the next chapter.

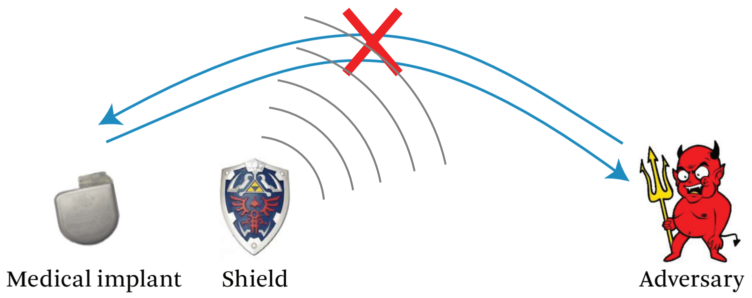


Figure 4.1 **Protecting an IMD without modifying it.** The shield jams any direct communication with the IMD. An authorized programmer communicates with the IMD only through the shield, with which it establishes a secure channel.

The shield actively prevents *any* device other than itself from communicating directly with the IMD. It does so by jamming messages sent to and from the IMD. Key to the shield's role is its ability to act as a jammer-cum-receiver, which enables it to jam the IMD's transmissions and prevent others from decoding them, while still being able to decode them itself. It also enables the shield to detect scenarios in which an adversary tries to overpower the shield's own transmissions to create a capture effect on the IMD and deliver an unauthorized message. By proxying IMD communications without requiring patients to interact directly with the shield, our design aligns with IMD industry trends toward wireless, time- and location-independent patient monitoring.

The next sections explain the jammer-cum-receiver's design, implementation, and use against passive and active adversaries.

4.4 Jammer-Cum-Receiver

A jammer-cum-receiver naturally needs to transmit and receive simultaneously. This section presents a design for such a full-duplex radio. Our design has two key features. First, it imposes no size restrictions and hence can be built as a small wearable device. Second, it cancels the jamming signal only at the device's receive antenna and at no other point in space—a necessary requirement for our application.

Our design, shown in Figure 4.2, uses two antennas: a jamming antenna and receive antenna. The jamming antenna transmits a random jamming signal. The receive antenna is simultaneously connected to both a transmit and a receive chain. The transmit chain sends an antidote signal that cancels the jamming signal at the receive antenna's front end, allowing the receive antenna to receive any signal without disruption from its own jamming signal.

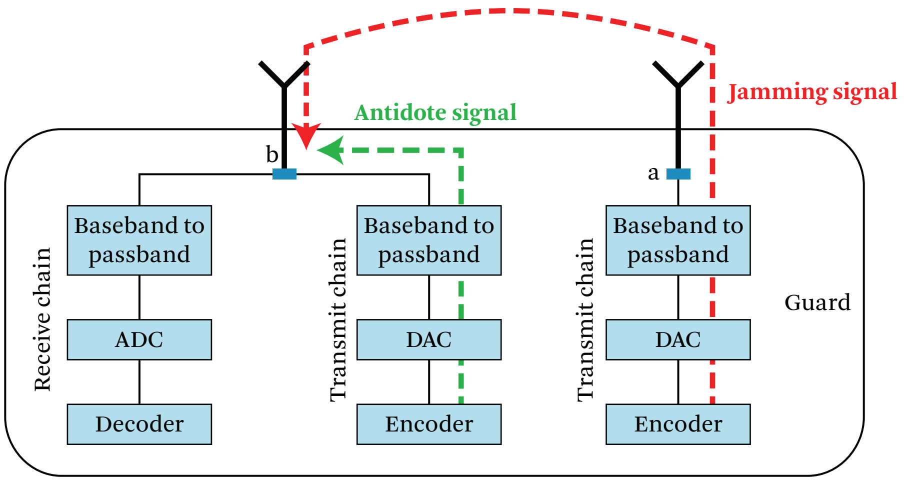


Figure 4.2 **The jammer-cum-receiver design uses two antennas.** A jamming antenna that transmits the jamming signal, and a receive antenna. The receive antenna is connected to both a transmit and receive chain. The antidote signal is transmitted from the transmit chain to cancel out the jamming signal in the receive chain.

The antidote signal can be computed as follows. Let $j(t)$ be the jamming signal, and let $x(t)$ be the antidote. Let H_{self} be the self-looping channel on the receive antenna (i.e., the channel from the transmit chain to the receive chain on the same antenna), and let $H_{jam\to rec}$ be the channel from the jamming antenna to the receive antenna. The signal received by the shield's receive antenna is

$$y(t) = H_{jam\to rec}\, j(t) + H_{self}\, x(t).$$

To cancel the jamming signal at the receive antenna, the antidote must satisfy

$$x(t) = -\frac{H_{jam\to rec}}{H_{self}}\, j(t). \tag{4.1}$$

Thus, by transmitting a random signal $j(t)$ on its jamming antenna and an antidote $x(t)$ on its receive antenna, the shield can receive signals transmitted by other nodes while jamming the medium.

Next, we show that the antidote cancels the jamming signal only at the shield's receive antenna, and no other location. Let $H_{jam\to l}$ and $H_{rec\to l}$ be the channels from the shield's jamming and receive antennas, respectively, to the adversary's location l. An antenna positioned at l receives the combined signal

$$
\begin{aligned}
y(t) &= H_{jam\to l}\, j(t) + H_{rec\to l}\, x(t) \\
&= (H_{jam\to l} - H_{rec\to l}\frac{H_{jam\to rec}}{H_{self}})\, j(t).
\end{aligned}
$$

For the jamming signal to be cancelled out at location l, the following must be satisfied:

$$\frac{H_{jam\to l}}{H_{rec\to l}} = \frac{H_{jam\to rec}}{H_{self}}.$$

Locating the shield's two antennas very close to each other ensures that at any location l the attenuation from the two antennas is comparable, i.e., $|\frac{H_{jam\to l}}{H_{rec\to l}}| \approx 1$ (see Chapter 7 in [Tse and Vishwanath 2005] for a detailed analysis). In contrast, $|\frac{H_{jam\to rec}}{H_{self}}| \ll 1$; $|H_{self}|$ is the attenuation on the short wire between the transmit and receive chains in the receive antenna, which is significantly less than the attenuation between the two antennas that additionally have to go on the air [Goldsmith 2005]. For example, in our USRP2 prototype, the ratio $|\frac{H_{jam\to rec}}{H_{self}}| \approx -27$ dB. Thus, the above condition is physically infeasible, and cancelling the jamming signal at the shield's receive antenna does not cancel it at any other location.

We note several ancillary properties of our design.

Transmit and receive chains connected to the same antenna. Off-the-shelf radios such as the USRP have both a receive and a transmit chain connected to the same antenna; they can in principle transmit and receive simultaneously on the same antenna. Traditional systems cannot exploit this property, however, because the transmit signal overpowers the receive chain, preventing the antenna from decoding any signal but its own transmission. When the jamming signal and the antidote signal cancel each other, the interference is cancelled and the antenna can receive from other nodes while transmitting.

Antenna cancellation vs. analog and digital cancellation. Cancelling the jamming signal with an antidote is a form of antenna cancellation. Thus, as in the antenna cancellation scheme by Choi et al. [2010], one can improve performance using hardware components such as analog cancelers [Radunovic et al. 2009]. In this case, the input to the analog canceler will be taken from points a and b in Figure 4.2; the output will be fed to the passband filter in the receive chain.

Channel estimation. Computing the antidote in Equation 4.1 requires knowing the channels H_{self} and $H_{jam\to rec}$. The shield estimates these channels using two methods. First, during a session with the IMD, the shield measures the channels immediately before it transmits to the IMD or jams the IMD's transmission. In the absence of an IMD session, the shield periodically (every 200 ms in our prototype) estimates this channel by sending a probe. Since the shield's two antennas are close to each other, the probe can be sent at a low power to allow other nodes to leverage spatial reuse to concurrently access the medium.

Wideband channels. Our discussion has been focused on narrowband channels. However, the same description can be extended to work with wideband channels which exhibit multipath effects. Specifically, such channels use OFDM, which divides the bandwidth into orthogonal subcarriers and treats each of the subcarriers as if it was an independent narrowband channel. Our model naturally fits in this context.[3]

4.5 Protecting Against Passive Eavesdroppers

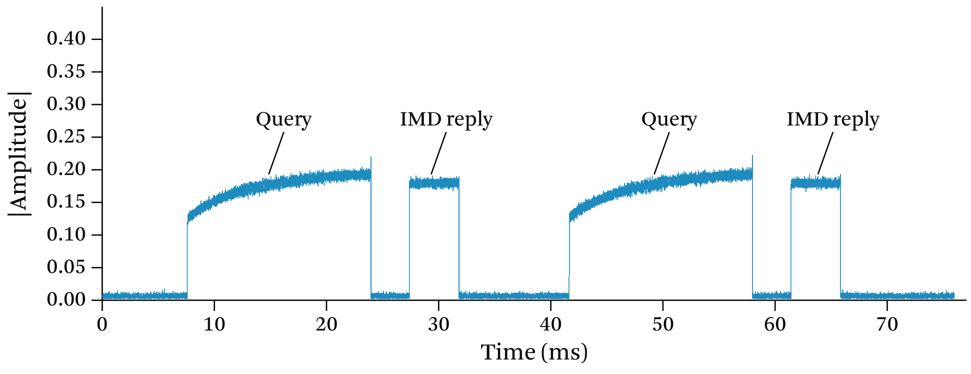
To preserve the confidentiality of an IMD's transmissions, the shield jams the IMD's signal on the channel. Since the wireless channel creates linear combinations of concurrently transmitted signals, jamming with a random signal provides a form of one-time pad, where only entities that know the jamming signal can decrypt the IMD's data [Shannon 1949]. The shield leverages its knowledge of the jamming signal and its jammer-cum-receiver capability to receive the IMD's data in the presence of jamming.

To realize our design goal, the shield must ensure that it jams every packet transmitted by the IMD. To this end, the shield leverages two properties of MICS-band IMDs [Federal Communications Commission 1999, International Telecommunications Union 1998].

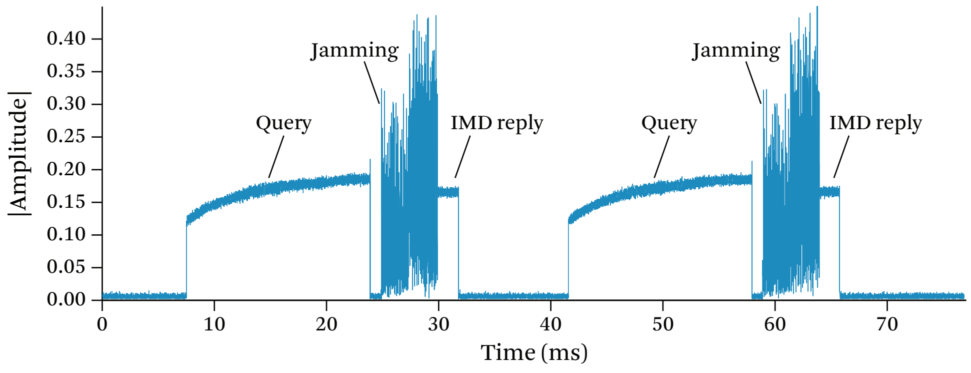
- An IMD does not transmit except in a response to a message from a programmer. The shield can listen for programmer transmissions and anticipate when the IMD may start transmitting.
- An IMD transmits in response to a message from a programmer without sensing the medium. This allows the shield to bound the interval during which the IMD replies after receiving a message.

Figure 4.3 shows an example exchange between a Medtronic Virtuoso implantable cardiac defibrillator (ICD) and a programmer (in this case, a USRP). Figure 4.3(a) shows that the Virtuoso transmits in response to a programmer's message after a fixed interval (3.5 ms). To check that the Virtuoso indeed does not sense the medium, we made

3. More generally, one could compute the multi-path channel and apply an equalizer [Gollakota et al. 2011] on the time-domain antidote signal that inverts the multi-path of the jamming signal.



(a) Without jamming



(b) With jamming

Figure 4.3 **Typical interaction between the Virtuoso IMD and its programmer.** Without jamming (a), the IMD transmits in response to an interrogation. The bottom graph (b) shows that the IMD transmits within a fixed interval without sensing the medium.

the programmer USRP transmit a message to the Virtuoso and within 1 ms transmit another random message. Figure 4.3(b) plots the resulting signal and shows that the Virtuoso still transmitted after the same fixed interval even though the medium was occupied.

Given the above properties, the shield uses the following algorithm to jam the IMD's transmissions. Let T_1 and T_2 be the lower and upper bounds on the time that the IMD takes to respond to a message, and let P be the IMD's maximum packet duration. Whenever the shield sends a message to the IMD, it starts jamming the medium exactly

T_1 ms after the end of its transmission. While jamming, the shield receives the signal on the medium using its receive antenna. The shield jams for $(T_2 - T_1) + P$ ms.

Additionally, to deal with scenarios in which the IMD may transmit in response to an unauthorized message, the shield uses its ability to detect active adversaries that might succeed at delivering a message to the IMD (see Section 4.6(d)). Whenever such an adversary is detected, the shield uses the same algorithm above, as if the message were sent to the IMD by the shield itself.

We note that each shield should calibrate the above parameters for its own IMD. In particular, for the IMDs tested in this work, the above parameters are as follows: $T_1 = 2.8$ ms, $T_2 = 3.7$ ms, and $P = 21$ ms.

Our design of the shield sets three sub-goals.

(a) Maximize Jamming Efficiency for a Given Power Budget. It is important to match the frequency profile of the jamming signal to the frequency profile of the jammed signal [Lopatka 1995]. To understand this issue, consider the example of the Virtuoso cardiac defibrillator. This device operates over a channel bandwidth of 300 KHz. However, it uses FSK modulation where a "0" bit is transmitted at one frequency f_0 and a "1" bit is transmitted at a different frequency f_1. Figure 4.4 shows the frequency profile of the FSK signal captured from a Virtuoso cardiac defibrillator. A jammer might create a jamming signal over the entire 300 KHz. However, since the frequency-domain representation of the received FSK signal has most of its energy concentrated around f_0 and f_1, an adversary can eliminate most of the jamming signal by applying two band-pass filters centered on f_0 and f_1.

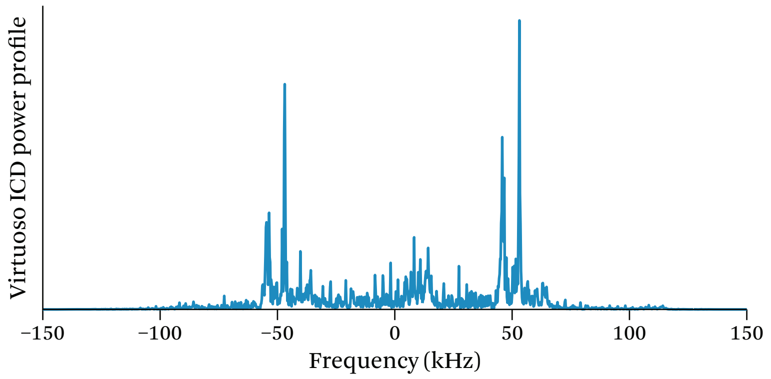


Figure 4.4 **The frequency profile of the FSK signal captured from a Virtuoso cardiac defibrillator** shows that most of the energy is concentrated around ±50 KHz.

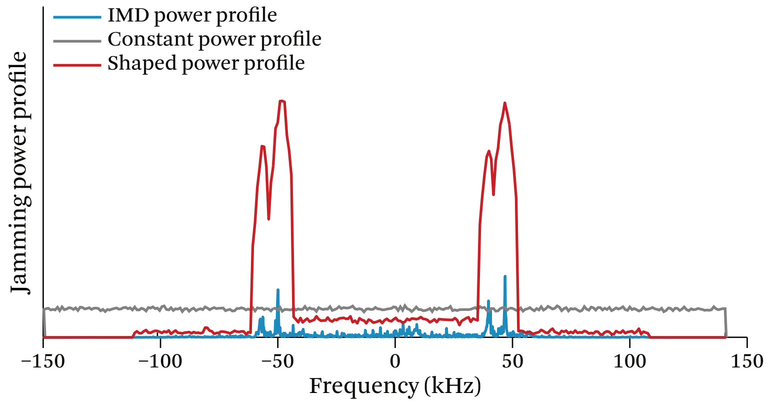


Figure 4.5 **Shaping the jamming signal's profile to match an IMD's** allows the shield to focus its jamming on frequencies that matter for decoding, as opposed to jamming across the entire 300 KHz channel.

Therefore, an effective jammer should consider the structure of the IMD's signal when crafting the jamming signal, shaping the amount of energy it puts in each frequency according to the frequency profile of the IMD signal. Figure 4.5 compares the power profile of a jamming signal that is shaped to fit the signal in Figure 4.4 and an oblivious jamming signal that uses a constant power profile. The figure shows that the shaped signal has increased jamming power in frequencies that matter for decoding.

To shape its jamming signal appropriately, the shield generates the jamming signal by taking multiple random white Gaussian noise signals and assigning each of them to a particular frequency bin in the 300 KHz MICS channel. The shield sets the variance of the white Gaussian noise in each frequency bin to match the power profile resulting from the IMD's FSK modulation in that frequency bin. We then take the IFFT of all the Gaussian signals to generate the time-domain jamming signal. This process generates a random jamming signal that has a power profile similar to the power profile generated by IMD modulation. The shield scales the amplitude of the jamming signal to match its hardware's power budget. The shield also compensates for any carrier frequency offset between its RF chain and that of the IMD.

(b) Ensure Independence of Eavesdropper Location. To ensure confidentiality, the shield must maintain a high bit error rate (BER) at the adversary, *independent* of the adversary's location. The BER at the adversary, however, strictly depends on its signal-to-interference-and-noise ratio, SINR_A [Goldsmith 2005]. To show that the BER at the adversary is independent of its location, we show that the SINR at the adversary is independent of its location.

Suppose the IMD transmits its signal at a power P_i dB and the shield transmits the jamming signal at a power P_j dB. The IMD's signal and the jamming signal will experience a pathloss to the adversary of L_i and L_j, respectively. Thus, the SINR at the adversary can be written in dB as

$$\text{SINR}_A = (P_i - L_i) - (P_j - L_j) - N_A, \tag{4.2}$$

where N_A is the noise in the adversary's hardware. Since Equation 4.2 is written in a logarithmic scale, the pathlosses translate into subtractions.

The pathloss from the IMD to the adversary can be expressed as the sum of the pathloss that the IMD's signal experiences in the body and on the air, i.e., $L_i = L_{body} + L_{air}$ [Panescu 2008]. Since the shield and the IMD are close together, the pathlosses they experience on the air to the adversary are approximately the same, i.e., $L_{air} \approx L_j$ [Tse and Vishwanath 2005]. Thus, we can rewrite Equation 4.2 as

$$\text{SINR}_A = (P_i - L_{body}) - P_j - N_A.$$

The above equation shows that SINR_A is independent of the adversary's location and can be controlled by setting the jamming power P_j to an appropriate value. This directly implies that the BER at the adversary is independent of its location.

(c) SINR Tradeoff between the Shield and the Adversary. Similarly to how we computed the SINR of an eavesdropper, we can compute the SINR of the shield (in dB) as

$$\text{SINR}_S = (P_i - L_{body}) - (P_j - G) - N_G,$$

where N_G is the thermal noise on the shield and G is the reduction in the jamming signal power at the receive antenna due to the antidote. The above equation simply states that SINR_S is the IMD power after subtracting the pathloss due mainly to in-body propagation, the residual of the jamming power $(P_j - G)$, and the noise.

Note that if one ignores the noise on the shield's receive antenna and the adversary's device (which are negligible in comparison to the other terms), one can express the relation between the two SINRs using a simple equation:

$$\text{SINR}_S = \text{SINR}_A + G. \tag{4.3}$$

This simplified view reveals an intrinsic tradeoff between the SINR at the shield and the adversary, and hence their BERs. To increase the BER at the adversary while maintaining a low BER at the shield, one needs to increase G, which is the amount of jamming power cancelled at the shield's receive antenna. We refer to G as the *SINR gap* between the shield and the adversary.

We show in Section 4.9.1 that for the tested IMDs, an SINR gap of $G = 32$ dB suffices to provide a BER of nearly 50% at the adversary (reducing the adversary to guessing) while maintaining reliable packet delivery at the shield.

4.6 Protecting Against Active Adversaries

Next, we explain our approach for countering active adversaries. At a high level, the shield detects unauthorized packets and jams them. The jamming signal combines linearly with the unauthorized signal, causing random bit flips during decoding. The IMD ignores these packets because they fail its checksum test.

The exact active jamming algorithm follows. Let S_{id} be an *identifying sequence*, i.e., a sequence of m bits that is always used to identify packets destined to the IMD. S_{id} includes the packets' physical-layer preamble and the subsequent header. When the shield is not transmitting, it constantly monitors the medium. If it detects a signal on the medium, it proceeds to decode it. For each newly decoded bit, the shield checks the last m decoded bits against the identifying sequence S_{id}. If the two sequences differ by fewer than a threshold number of bits, b_{thresh}, the shield jams the signal until the signal stops and the medium becomes idle again.

The shield also uses its receive antenna to monitor the medium while transmitting. However, in this case, if it detects a signal concurrent to its transmission, it switches from transmission to jamming and continues jamming until the medium becomes idle again. The reason the shield jams any concurrent signal without checking for S_{id} is to ensure that an adversary cannot successfully alter the shield's own message on the channel in order to send an unauthorized message to the IMD.

We note five subtle design points:

(a) Choosing Identifying Sequences. Our algorithm relies on the identifying sequence S_{id} in order to identify transmissions destined for the protected IMD. We therefore desire a method of choosing a per-device S_{id} based on unique device characteristics. Fortunately, IMDs already bear unique identifying characteristics. For example, the Medtronic IMDs that we tested (the Virtuoso ICD and the Concerto CRT) use FSK modulation, a known preamble, a header, and the device's ID, i.e., its 10-byte serial number. More generally, each wireless device has an FCC ID, which allows the designer to look up the device in the FCC database and verify its modulation, coding, frequency, and power profile.[4] One can use these specifications to choose an appropriate identifying sequence. Furthermore, once in a session, the IMD locks on to a unique channel, to receive any future commands. Since other IMD–programmer pairs avoid occupied channels, this channel ID can be used to further specify the target IMD.

(b) Setting the Threshold b_{thresh}. If an adversary can transmit a signal and force the shield to experience a bit error rate higher than the IMD's, it may prevent the shield from jamming an unauthorized command that the IMD successfully decodes and executes. However, we argue that such adversarial success is unlikely, for two reasons. First, because the signal goes through body tissue, the IMD experiences an additional pathloss that could be as high as 40 dB [Sayrafian-Pour et al. 2010], and hence it naturally experiences a much weaker signal than the shield. Second, the IMD uses a harder constraint to accept a packet than the constraint the shield uses to jam a packet. Specifically, the IMD requires that all bits be correct to pass a checksum, while the shield tolerates some differences (up to b_{thresh} bits) between the identifying sequence and the received one. We describe our empirical method of choosing b_{thresh} in Section 4.9.1(c).

(c) Customizing for the MICS Band. It is important to realize that the shield can listen to the entire 3 MHz MICS band, transmit in all or any subset of the channels in this band, and further continue to listen to the whole band as it is transmitting in any subset of the channels. It is fairly simple to build such a device by making the radio front end as wide as 3 MHz and equipping the device with per-channel filters. This enables the shield to process the signals from all channels in the MICS band simultaneously.

The shield uses this capability to monitor the entire 3 MHz MICS band because an adversary can transmit to the IMD on any channel in the band. This monitoring allows the shield to detect and counter adversarial transmissions even if the adversary uses frequency hopping or transmits in multiple channels simultaneously to try to confuse the shield. The shield jams any given 300 KHz channel if the channel contains a signal that matches the constraints described in the active jamming algorithm.

(d) Complying with FCC Rules. The shield must adhere to the FCC power limit even when jamming an adversary. However, as explained in Section 4.2, a sophisticated adversary may use a transmission power much higher than the FCC limit. In such cases, the adversary will be able to deliver its packet to the IMD despite jamming. However, the shield is still useful because it can detect the high-powered adversary in real time and raise an alarm to attract the attention of the patient or a caregiver. Such alarms may be

4. FCC ID number search can be conducted at http://www.fcc.gov/searchtools.html. For example, the FCC ID LF5MICS refers to Medtronic IMDs we tested.

similar to a cell phone alarm, i.e., the shield may beep or vibrate. It is desirable to have a low false positive rate for such an alarm. To that end, we calibrate the shield with an IMD to find the minimum adversarial transmit power that can trigger a response from the IMD despite jamming. We call this value P_{thresh}. When the shield detects a potentially adversarial transmission, it checks whether the signal power exceeds P_{thresh}, in which case it raises an alarm.

Finally, we note that when the shield detects a high-powered active adversary, it also considers the possibility that the adversary will send a message that triggers the IMD to send its private data. In this case, the shield applies the passive jamming algorithm: in addition to jamming the adversary's high-powered message, it jams the medium afterward, as detailed in Section 4.5.

(e) Battery Life of the Shield. Since jamming consumes power, one may wonder how often the shield needs to be charged. In the absence of attacks, the shield jams only the IMD's transmissions, and hence transmits approximately as often as the IMD. IMDs are typically nonrechargeable power-limited devices that do not transmit frequently [Falcon 2004]. Thus, in this mode of operation, we do not expect the battery of the shield to be an issue. When the IMD is under an active attack, the shield will have to transmit as often as the adversary. However, since the shield transmits at the FCC power limit for the MICS band, it can last for a day or longer even if transmitting continuously. For example, wearable heart rate monitors that continuously transmit ECG signals can last 24–48 h.[5]

4.7 Implementation

We implement a proof-of-concept prototype shield with GNU Radio and USRP2 hardware. The prototype uses the USRP's RFX400 daughterboards, which operate in the MICS band [Federal Communications Commission 1999]. The USRP2 does not support multiple daughterboards on the same motherboard, so we implement a two-antenna shield with two USRP2 radio boards connected via an external clock[6] so that they act as a single node. The two antennas are placed right next to each other. Our design for a two-antenna jammer-cum-receiver requires the receive antenna to be always connected to both a transmit and a receive chain. To enable the shield's receive antenna to transmit and receive simultaneously, we turn off the USRP RX/TX switch, which leaves both the transmit and receive chains connected to the antenna all the time.

5. Zephyr Inc. BioHarness http://www.zephyranywhere.com/products/bioharness-3/; last retrieved May 2014.

6. Jackson Labs Fury GPSDO. Available from http://www.jackson-labs.com/; last retrieved May 2014.

Specifically, we set `atr_txval=MIX_EN` and `atr_rxval=ANT_SW` in the TX chain, and we set `atr_txval=MIX_EN` and `atr_rxval=MIX_EN` in the RX chain, in the USRP2's firmware and FPGA code. Finally, we equip the shield with FSK modulation and demodulation capabilities so that it can communicate with an IMD.

4.8 Testing Environment

Our experiments use the following devices:

- Medtronic Virtuoso DR implantable cardiac defibrillators (ICDs);
- a Medtronic Concerto cardiac resynchronization therapy device (CRT);
- a Medtronic Vitatron Carelink 2090 Programmer; and
- USRP2 software radio boards.

In our *in vitro* experiments, the ICD and CRT play the role of the protected IMD. The USRP devices play the roles of the shield, the adversary, and legitimate users of the MICS band. We use the programmer off-line with our active adversary; the adversary records the programmer's transmissions in order to replay them later. Analog replaying of these captured signals doubles their noise, reducing the adversary's probability of success, so the adversary demodulates the programmer's FSK signal into the transmitted bits to remove the channel noise. The adversary then re-modulates the bits to obtain a clean version of the signal to transmit to the IMD.

Figure 4.6 depicts the testing setup. To simulate implantation in a human, we followed prior work [Halperin et al. 2008c] and implanted each IMD beneath 1 cm of bacon, with 4 cm of 85% lean ground beef packed underneath. We placed the shield next to the IMD on the bacon's surface to simulate a necklace. We varied the adversary's location between 20 cm and 30 m, as shown in the figure.

4.9 Evaluation

We evaluate our prototype of a shield against commercially available IMDs. We show that the shield effectively protects the confidentiality of the IMD's messages and defends the IMD against commands from unauthorized parties. We experiment with both the Virtuoso ICD and the Concerto CRT. However, since the two IMDs did not show any significant difference, we combine the experimental results from both devices and present them together. Our results can be summarized as follows.

- In practice, our antenna cancellation design can cancel about 32 dB of the jamming signal at the receive antenna (Section 4.9.1(a)). This result shows that our

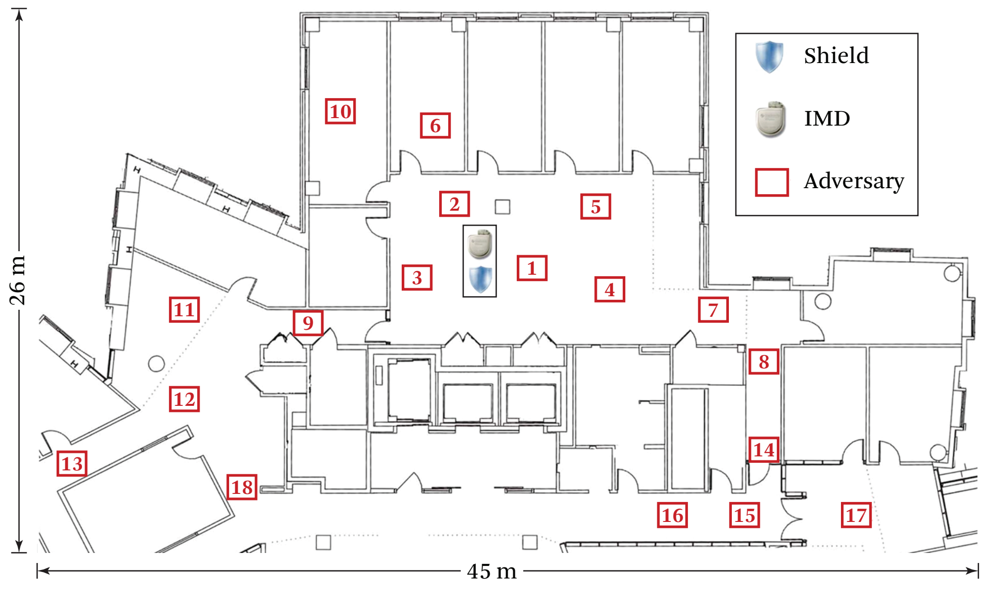


Figure 4.6 **Testbed setup showing shield, IMD, and adversary locations.** We experiment with 18 adversary locations, numbered here in descending order of received signal strength at the shield.

design achieves similar performance to the antenna cancellation algorithm proposed in prior work [Choi et al. 2010], but without requiring a large antenna separation.

- Setting the shield's jamming power 20 dB higher than the IMD's received power allows the shield to achieve a high bit error rate at adversarial locations while still being able to reliably decode the IMD's transmissions (Section 4.9.1(b)). The shield's increased power still complies with FCC rules in the MICS band since the transmit power of implanted devices is 20 dB less than the transmit power for devices outside the body [PCTEST Engineering Labs, Inc. 2002, PCTEST Engineering Labs, Inc. 2007].
- With the above setting, the bit error rate at a passive eavesdropper is nearly 50% at all tested locations, i.e., an eavesdropping adversary's decoding efforts are no more effective than random guessing. Further, even while jamming, the shield can reliably decode the IMD's packets with a packet loss rate less than 0.2%. We conclude that the shield and the IMD share a channel inaccessible to other parties (Section 4.9.2).

- When the shield is present and active, an adversary using off-the-shelf IMD programmers cannot elicit a response from the protected IMD even from distances as small as 20 cm. A more sophisticated adversary that transmits at 100 times the shield's power successfully elicits IMD responses only at distances less than 5 m, and only in line-of-sight locations. Further, the shield detects these high-powered transmissions and raises an alarm. We conclude that the shield significantly raises the bar for such high-powered adversarial transmissions (Section 4.9.3).

4.9.1 Micro-Benchmark Results

In this section, we calibrate the parameters of the shield and examine the performance of its components.

(a) Antenna Cancellation. We first evaluate the performance of the antenna cancellation algorithm in Section 4.4, in which the shield sends an antidote signal to cancel the jamming signal on its receive antenna.

In this experiment, the shield transmits a random signal on its jamming antenna and the corresponding antidote on its receive antenna. In each run, it transmits 100 Kb without the antidote, followed by 100 Kb with the antidote. We compute the received power at the receive antenna with and without the antidote. The difference in received power between the two trials is the jamming cancellation resulting from the transmission of the antidote.

Figure 4.7 shows the CDF of the amount of cancellation over multiple runs of the experiment. It shows that the average reduction in jamming power is about 32 dB. The figure also shows that the variance of this value is small. This result shows that the antenna cancellation algorithm introduced in this work achieves similar performance

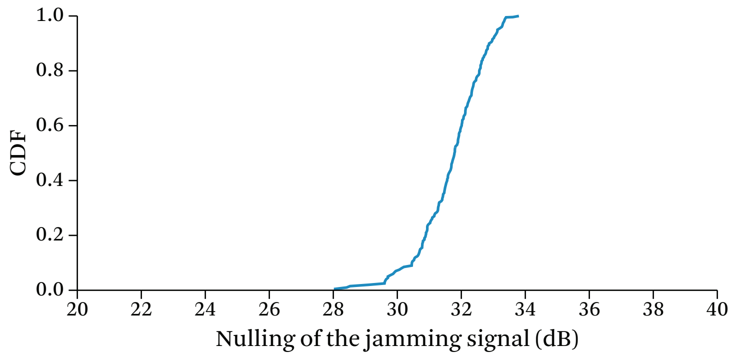


Figure 4.7 **Antenna cancellation.** The antidote signal reduces the jamming signal by 32 dB on average.

to the algorithm proposed by Choi et al. [Choi et al. 2010], but without requiring a large antenna separation.[7]

(b) Tradeoffs between Eavesdropper Error and Shield Error. The aforementioned 32 dB of cancellation at the shield's receive antenna naturally sets an upper bound on the jamming power: if the residual error after jamming cancellation is too high, the shield will fail to decode the IMD's data properly.

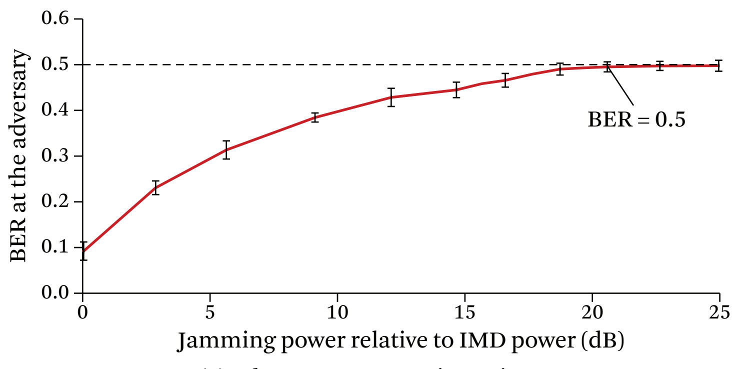
To explore the tradeoff between the error at the shield and the error at an eavesdropper, we run the following experiment. We place the IMD and the shield at their marked locations in Figure 4.6, and we place a USRP eavesdropper 20 cm away from the IMD at location 1. In each run of the experiment, the shield repeatedly triggers the IMD to transmit the same packet. The shield also uses its jammer-cum-receiver capability to simultaneously jam and decode the IMD's packets. The eavesdropper tries to decode the IMD packets, in the presence of jamming, using an optimal FSK decoder [Meyr et al. 1998].

Figure 4.8(a) plots the eavesdropper's BER as a function of the shield's jamming power. Since the required jamming power naturally depends on the power of the jammed IMD's signal, the x-axis reports the shield's jamming power relative to the power of the signal it receives from the IMD. The figure shows that if the shield sets its jamming power 20 dB higher than the power of the signal it receives from the IMD, the eavesdropper's BER is 50%. Thus the eavesdropper's decoding task is no more successful than random guessing.

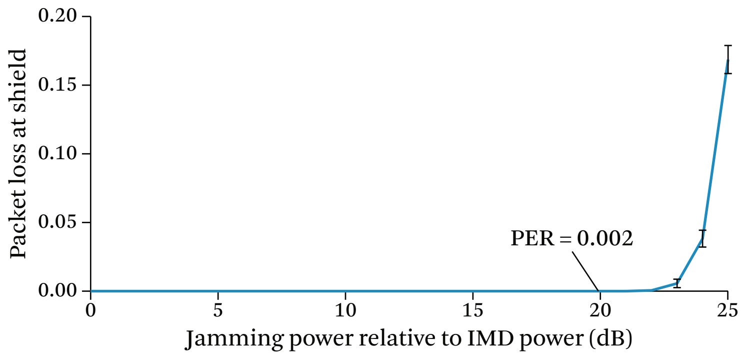
Next, we check that the above setting allows the shield to reliably decode the IMD's packets. As above, Figure 4.8(b) plots the shield's packet loss rate as a function of its jamming power relative to the power of the signal it receives from the IMD. The figure shows that if the shield's jamming power is 20 dB higher than the IMD's power, the packet loss rate is no more than 0.2%. We conclude that this jamming power achieves both a high error rate at the eavesdropper and reliable decoding at the shield.

We note that the shield's increased power, described above, still complies with FCC rules on power usage in the MICS band because the transmit power of implanted devices is 20 dB less than the maximum allowed transmit power for devices outside the body [PCTEST Engineering Labs, Inc. 2002, PCTEST Engineering Labs, Inc. 2007].

7. Choi et al. [2010] also combine antenna cancellation with analog and digital cancellation to obtain a total cancellation of 60 dB at the receive antenna. However, we show in Section 4.9.2 that for our purposes, a cancellation of 32 dB suffices to achieve our goal of high reliability at the shield and nearly 50% BER at the adversary.



(a) Adversary's BER vs. jamming power



(b) Shield's PER vs. jamming power

Figure 4.8 **Tradeoff between BER at the eavesdropper and reliable decoding at the shield.** If the shield sets its jamming power 20 dB higher than the power it receives from the IMD, it can ensure that an eavesdropper sees a BER around 50% (a)—effectively reducing the eavesdropper to guessing—while keeping the packet loss rate (PER) at the shield as low as 0.2% (b).

(c) Setting the Jamming Parameters. Next, we calibrate the jamming parameters for countering active adversaries. The shield must jam unauthorized packets sent to the IMD it protects. It must jam these packets even if it receives them with some bit errors, because they might otherwise be received correctly at the IMD. We therefore empirically estimate an upper bound, b_{thresh}, on the number of bit flips an IMD accepts in an adversary's packet header. The shield uses this upper bound to identify packets that must be jammed.

To estimate b_{thresh}, we perform the following experiment. First, a USRP transmits unauthorized commands to the IMD to trigger it to send patient data. We repeat the

Table 4.1 **Adversarial RSSI that elicits IMD responses despite the shield's jamming**

P_{thresh}: Adversary power that elicits IMD response	Minimum	−11.1 dBm
	Average	−4.5 dBm
	Standard Deviation	3.5 dBm

experiment for all locations in Figure 4.6. The shield stays in its marked location in Figure 4.6, but its jamming capability is turned off. However, the shield logs all of the packets transmitted by the IMD as well as the adversarial packets that triggered them. We process these logs offline and, for packets that successfully triggered an IMD response despite containing bit errors, we count the number of bit flips in the packet header. Our results show that it is unlikely that a packet will have bit errors at the shield but still be received correctly by the IMD. Out of 5000 packets, only 3 packets showed errors at the shield but still triggered a response from an IMD. The maximum number of bit flips in those packets was 2, so we conservatively set $b_{thresh} = 4$.

Next, we measure P_{thresh}, the minimum adversary RSSI at the shield that can elicit a response from the IMD in the presence of jamming. To do so, we fix the location of the IMD and the shield as shown in Figure 4.6. Again we use a USRP that repeatedly sends a command to trigger the IMD to transmit. We fix the adversary in location 1 and vary its transmit power. Table 4.1 reports the minimum and average RSSI at the shield's receive antenna for all packets that succeeded in triggering the IMD to transmit. We set P_{thresh} 3 dB below the minimum RSSI in the table and use that value for all subsequent experiments.

4.9.2 Protecting from Passive Adversaries

To evaluate the effectiveness of the shield's jamming, we run an experiment in which the shield repeatedly triggers the IMD to transmit the same packet. The shield also uses its jammer-cum-receiver capability to jam the IMD's packets while it decodes them. We set the shield's jamming power as described in Section 4.5. In each run, we position an eavesdropper at a different location shown in Figure 4.6 and make the IMD send 1000 packets. The eavesdropping adversary attempts to decode the IMD's packets using an optimal FSK decoder [Meyr et al. 1998]. We record the BER at the eavesdropper and the packet loss rate at the shield.

Figure 4.9 plots a CDF of the eavesdropper's BER taken over all locations in Figure 4.6. The CDF shows that the eavesdropper's BER is nearly 50% in all tested locations. We conclude that our design of the shield achieves the goal of protecting the confiden-

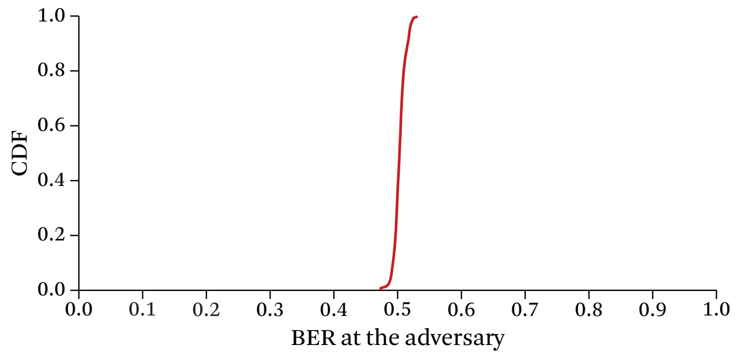


Figure 4.9 **CDF of an eavesdropper's BER over all eavesdropper locations in Figure 4.6.** At all locations, the eavesdropper's BER is nearly 50%, which makes its decoding task no more successful than random guessing. The low variance in the CDF shows that an eavesdropper's BER is independent of its location.

tiality of IMD's transmissions from an eavesdropper regardless of the eavesdropper's location.

For the same experiment, Figure 4.10 plots a CDF of the packet loss rate of IMD-transmitted packets at the shield. Each point on the *x*-axis refers to the packet loss rate over 1000 IMD packets. The average packet loss rate is about 0.2%, considered low for wireless systems [Eckhardt and Steenkiste 1996]. Such a low loss rate is due to two factors. First, we locate the shield fairly close to the IMD, so it receives the IMD's signal at a relatively high SNR. Second, the jamming cancellation is sufficient to maintain a high SNR that ensures a low packet loss rate. We conclude that the shield can decode the IMD's packets, even while jamming.

4.9.3 Protecting from Active Adversaries

We distinguish between two scenarios representing different levels of adversarial sophistication. In the first, we consider scenarios in which the adversary uses an off-the-shelf IMD programmer to send unauthorized commands to the IMD. In the second, a more sophisticated adversary reverse-engineers the protocol and uses custom hardware to transmit with much higher power than is possible in the first scenario.

(a) Adversary That Uses a Commercial IMD Programmer. The simplest way an adversary can send unauthorized commands to an IMD is to obtain a standard IMD programmer and use its built-in radio. Since commercial programmers abide by FCC rules, in this scenario, the adversary's transmission power will be comparable to that of the shield.

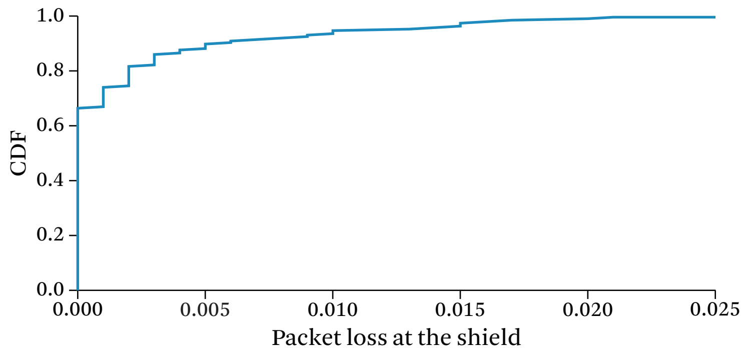


Figure 4.10 **Packet loss at the shield.** When the shield is jamming, it experiences an average packet loss rate of only 0.2% when receiving the IMD's packets. We conclude that the shield can reliably decode the IMD's transmissions despite jamming.

Using an IMD programmer we obtained via a popular auction website, we play the role of such an active adversary. We use the setup in Figure 4.6, fixing the IMD's and shield's locations and transmitting unauthorized commands from all the marked locations. As shown in the figure, we experiment with both line-of-sight and non-line-of-sight locations as well as nearby (20 cm) and relatively far locations (30 m).

To test whether the shield's jamming is effective against unauthorized commands, regardless of which unauthorized command the adversary chooses to send, we experiment with two types of adversarial commands: those that trigger the IMD to transmit its data with the objective of depleting its battery, and those that change the IMD's therapy parameters. In each location, we play each command 100 times with the shield on and 100 times with the shield off. After each attempt, we check whether the command was successful. To determine whether the first type of command was successful—i.e., whether it elicited a reply—we sandwiched a USRP observer along with the IMD between the two slabs of meat. To allow the USRP observer to easily check whether the IMD transmitted in response to the adversary's command, we configure the shield to jam only the adversary's packets, not the packets transmitted by the IMD. To determine whether a therapy modification command was successful, we use the IMD programmer to read the therapy parameters before and after the attempt.

Figures 4.11 and 4.12 show the results of these experiments. They plot the probability that adversarial commands succeed with the shield off (absent) and on (present), each as a function of adversary locations. The locations are ordered by decreasing SNR at the USRP observer. The figures show the following:

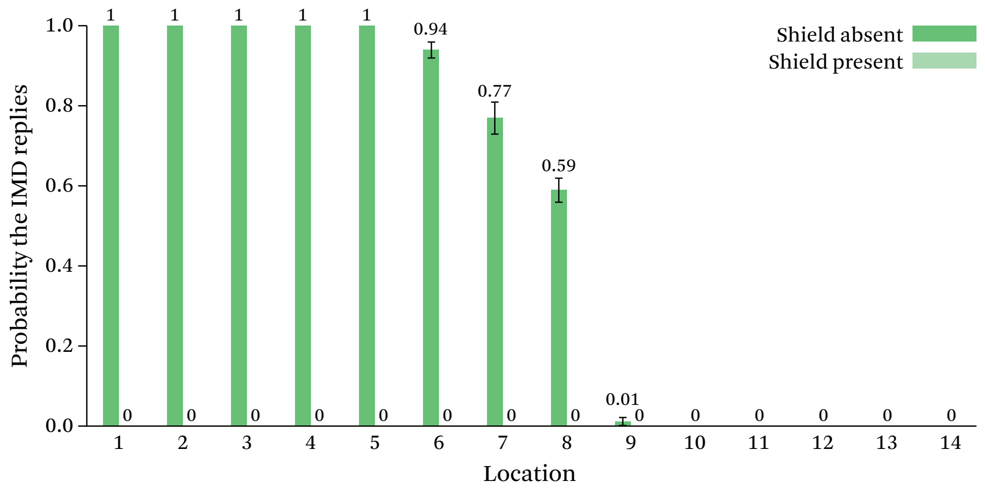


Figure 4.11 **Without the shield triggering an IMD to transmit and deplete its battery using an off-the-shelf IMD programmer succeeds with high probability.** With the shield, such attacks fail.

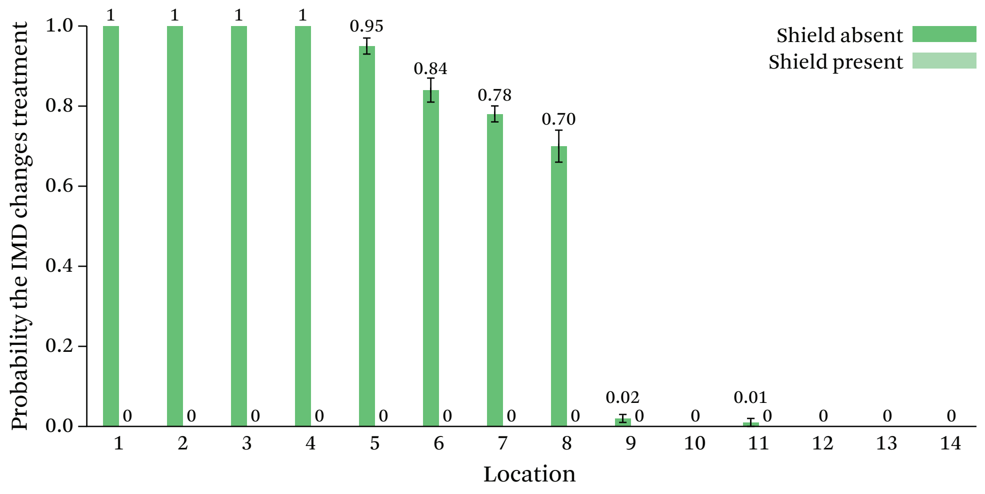


Figure 4.12 **Without the shield, an adversary using an off-the-shelf programmer to send unauthorized commands (in this case, to modify therapy parameters) succeeds with high probability.** The shield materially decreases the adversary's ability to control the IMD.

- When the shield is off, adversaries located up to 14 m away (location 8) from the IMD—including non-line-of-sight locations—can change the IMD's therapy parameters or cause the IMD to transmit its private data using precious battery energy, in contrast to past work in which the adversarial range is limited to a few centimeters [Halperin et al. 2008c]. We attribute this increased adversarial range to recent changes in IMD design that enable longer-range radio communication (MICS band) meant to support remote monitoring and a larger sterile field during surgery.
- When the shield is on, it successfully prevents the IMD from receiving adversarial commands as long as the adversary uses a device that obeys FCC rules on transmission power—even when the adversary is as close as 20 cm.
- There is no statistical difference in success rate between commands that modify the patient's treatment and commands that trigger the IMD to transmit private data and deplete its battery.

(b) High-Powered Active Adversary. Next, we experiment with scenarios in which the adversary uses custom hardware to transmit at 100 times the shield's transmit power. The experimental setup is similar to those discussed above; specifically, we fix the locations of the IMD and the shield and vary the high-powered adversary's position among the numbered locations in Figure 4.6. Each run has two phases: one with the shield off and another with the shield on. Since we found no statistical difference in success rate between unauthorized commands that trigger the IMD to transmit and those that change its therapy parameters, we show results only for the therapy modification command.

Figure 4.13 shows the results of this experiment in terms of the observed probability of adversarial success, with the shield both on and off. It also shows the observed probability that the shield raises an alarm, which is how the shield responds to a high-powered (above P_{thresh}) adversarial transmission. The figure further shows the following.

- When the shield is off, the adversary's increased power allows it to elicit IMD responses from as far as 27 m (location 13) and from non-line-of-sight locations.
- When the shield is on, the adversary elicits IMD responses only from close locations. Thus, the shield's presence raises the bar even for high-powered adversaries.
- Whenever the adversary elicits a response from the IMD in the presence of the shield, the shield raises an alarm. The shield also raises an alarm in response to

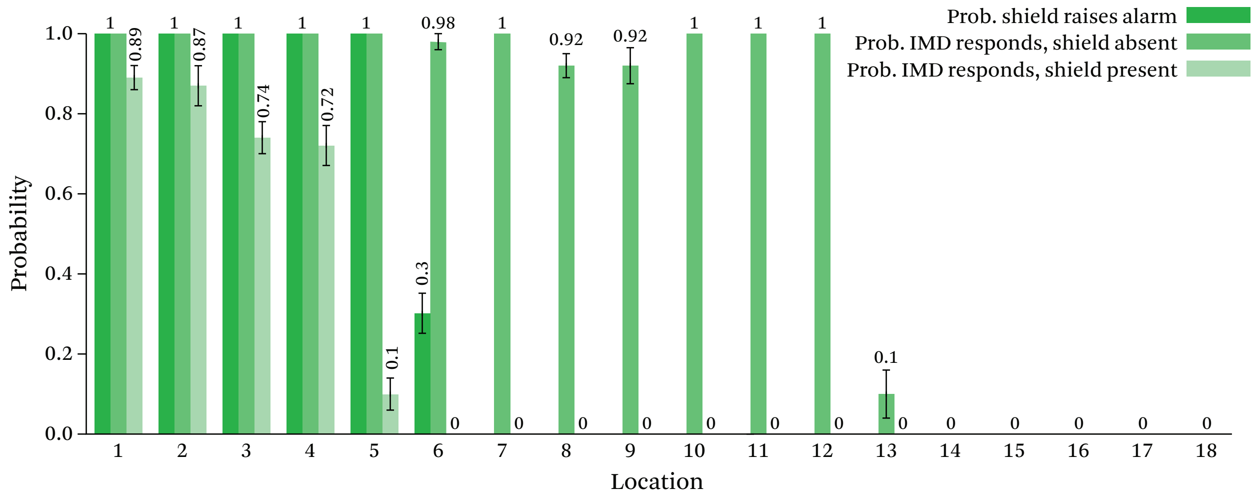


Figure 4.13 **High-powered adversary.** Without the shield, an adversary transmitting at 100 times the shield's power can change the IMD's therapy parameters even from non-line-of-sight locations up to 27 m away. With the shield, the adversary is successful only from line-of-sight locations less than 5 m away, and the shield raises an alarm.

unsuccessful adversarial transmissions that are high powered and emanate from nearby locations (e.g., location 6). While this conservative alert results in false positives, we believe it is reasonable to alert the patient that an adversary is nearby and may succeed at controlling the IMD.

4.10 Coexistence

We investigate how the presence of a shield affects other legitimate users of the medium. As explained in Section 4.1, the FCC rules for medical devices in the MICS band require such devices to monitor a candidate channel for 10 ms and avoid using occupied channels. As a result, two pairs of honest medical devices are unlikely to share the same 300 KHz channel. We focus our evaluation on coexistence with the meteorological devices that are the primary users of the MICS band (and hence can transmit even on occupied channels).

In this experiment, we position the IMD and the shield in the locations marked on Figure 4.6. We make a USRP board alternate between sending unauthorized commands to the IMD and transmitting cross-traffic unintended for the IMD. The cross-traffic is modeled after the transmissions of meteorological devices, in particular a Vaisala digital radiosonde RS92-AGP [Åkerberg 2004] that uses GMSK modulation. For

Table 4.2 **Coexistence results.** Jamming behavior and turn-around time in the presence of simulated meteorological cross-traffic.

Probability of Jamming	Cross-Traffic	0
	Packets that trigger IMD	1
Turn-around Time	Average	270 μs
	Standard Deviation	23 μs

each of the adversary positions in Figure 4.6, we make the USRP alternate between one packet to the IMD and one cross-traffic packet. The shield logs all packets it detects and reports which of them it jammed.

Post-processing of the shield's log showed that the shield did not jam any of the cross-traffic packets, regardless of the transmitter's location. In contrast, the shield jammed all of the packets that it detected were addressed to the IMD; see Table 4.2. Further, our software radio implementation of the shield takes $270 \pm 23\ \mu$s after an adversary stops transmitting to turn around and stop its own transmissions. This delay is mainly due to the shield's being implemented in software. A hardware implementation would have a more efficient turn-around time of tens of microseconds. (Note, for example, that a 802.11 card can turn around in a SIFS duration of 10 μs.) The low turn-around time shows that the shield does not continuously jam the medium (thereby denying others access to it).

4.11 Related Work

Recent innovations in health-related communication and networking technologies range from low-power implantable radios that harvest body energy [Koplow et al. 2008] to medical sensor networks for in-home monitoring and diagnosis [Shnayder et al. 2005]. Past work has also studied the vulnerabilities of these systems and proposed new designs that could improve their security [Halperin et al. 2008b, Halperin et al. 2008c]. Our work builds on this foundation, but it differs from all past works in that it presents the first system that defends existing commercial IMDs against adversaries who eavesdrop on transmissions or send unauthorized commands.

Our design is motivated by the work of Halperin et al., who analyzed the security properties of an implantable cardiac device and demonstrated its vulnerability to adversarial actions that compromise data confidentiality or induce potentially harmful heart rhythms [Halperin et al. 2008b, Halperin et al. 2008c]. They also suggested adding passively powered elements to implantable devices to allow them to authenticate their interlocutors. Along similar lines, Denning et al. propose a class of devices called *cloakers* that would share secret keys with IMDs [Denning et al. 2008]; an IMD would attempt to detect an associated cloaker's presence either periodically or when presented with an unknown programmer. Unlike these three proposals, our technique does not require cryptographic methods and is directly applicable to IMDs that are already implanted.

Other work has focused on the problem of key distribution for cryptographic security. Cherukuri et al. propose using consistent human biometric information to generate identical secret keys at different places on a single body [Cherukuri et al. 2003]. Schechter suggests that key material could be tattooed onto patients using ultraviolet micro-pigmentation [Schechter 2010].

Our work builds on a rich literature in wireless communication. Specifically, past work on jamming focuses on enabling wireless communication in the presence of adversarial jamming [Liu et al. 2010, Pöpper et al. 2009]. Some past work, however, has proposed to use friendly jamming to prevent adversarial access to RFID tags, sensor nodes, and IMDs [Martinovic et al. 2009, Rieback et al. 2005, Xu et al. 2011]. Our work is complementary to this past work but differs from it in that our jammer can transmit and receive at the same time; this allows it to decode IMD messages while protecting their confidentiality.

Our work is related to prior work on physical-layer information-theoretic security. Past work in this area has shown that if the channel to the receiver is better than the channel to an eavesdropper, the sender–receiver pair can securely communicate [Csiszar and Korner 1978, Siavoshani et al. 2011, Wyner 1975]. Also, our prior work proposes iJam, an OFDM-based technique that jams while receiving to prevent unauthorized receivers from obtaining a protected signal [Gollakota and Katabi 2011]. However, iJam is not applicable to IMDs, because it relies on the intrinsic characteristics of OFDM signals, which differ greatly from IMDs' FSK signals. Further, iJam requires changes to both the transmitter and receiver, and hence does not immediately apply to IMDs that are already implanted.

Finally, our work also builds on past work on full-duplex radio [Choi et al. 2010, Duarte and Sabharwal 2010]. Ours, however, differs from all past works in that it is the first to demonstrate the value of using full-duplex radios for security. Furthermore, we implement a radio where the antennas are placed next to each other so that it can be built as a small device and show both empirically and analytically that our design secures IMDs using only 30 dB cancellation which is significantly less than the 60–80 dB cancellation required by prior work [Duarte and Sabharwal 2010, Choi et al. 2010].

4.12 Discussion

This book addresses the problem of communication security for implanted medical devices. The key challenge in addressing this problem stems from the difficulty of modifying or replacing implanted devices. We present the design and implementation of a wireless physical-layer solution that delegates the task of protecting IMD communication to an external device called the shield. Our evaluation shows that the shield effectively provides confidentiality for IMDs' transmitted data and shields IMDs from unauthorized commands, both without requiring any changes to the IMDs themselves.

More generally, the influx of wireless communication in medical devices brings a number of domain-specific problems that require the expertise of both the wireless and security communities. IMDShield provides a case study for how one can leverage expertise in wireless device design to secure medical implants. Beyond IMDShield, we believe that this inter-disciplinary approach can enable novel domain-specific solutions that address the security and privacy challenges in the medical domain.

5 Secure Pairing without Passwords or Prior Secrets

Recent trends in the security of home WiFi networks are driven by two phenomena: ordinary users often struggle with the security setup of their home networks [Kelton Research 2006], and, as a result, some of them end up skipping security activation [WiFi Alliance 2006, Norman 2009]. Simultaneously, there is a proliferation of WiFi gadgets and sensors that do not support an interface for entering a key. These include WiFi sound systems, medical sensors, USB keys, light and temperature sensors, motion detectors and surveillance sensors, home appliances, and game consoles. Even new models of these devices are unlikely to support a keypad because of limitations on their form factor, style, cost, or functionality. Responding to these two requirements—easing security setup for home users and securing devices that do not have an interface for entering a key—the WiFi Alliance has introduced the Push Button Configuration (PBC) mechanism [WiFi Alliance 2006]. To establish a secure connection between two WiFi devices, the user pushes a button on each device, and the devices broadcast their Diffie–Hellman public keys [Diffie and Hellman 1976], which they then use to protect all future communication. PBC is a *mandatory* part of the new WiFi Protected Setup certification program.[1] It is already adopted by the major WiFi manufacturers (e.g., Cisco, NetGear, HP, Microsoft, Sony) and implemented in about 2000 new products from 117 different companies.[2]

Unfortunately, the PBC approach taken by the WiFi Alliance does not fully address WiFi security. Diffie–Hellman's key-exchange protocol [Diffie and Hellman 1976] protects against only passive adversaries that snoop on the wireless medium to obtain key

1. "WiFi Alliance to ease setup of home WiFi networks with new industry wide program," January 2007. Available from http://www.wi-fi.org/news-events/newsroom/wi-fi-alliance-to-ease-setup-of-home-wi-fi-networks-with-new-industry-wide; last retrieved May 2014.

2. http://www.wi-fi.org; last retrieved May 2014.

exchange messages. Since the key exchange messages are not authenticated in any way, the protocol is vulnerable to an active man-in-the-middle (MITM) attack. That is, an adversary can impersonate each device to the other, convincing both devices to establish a secure connection via the adversary. With WiFi increasingly used in medical sensors that transmit a patient's vital signals [Halamka 2010] and surveillance sensors that protect one's home, there is a concern that, being vulnerable to MITM attacks, PBC may give users a false sense of security [WiFi Alliance 2006, Kuo et al. 2007].

One may wonder why the WiFi Alliance did not adopt a user-friendly solution that also protects against MITM attacks. We believe the reason is that existing user-friendly solutions to MITM attacks require devices to support an out-of-band communication channel [Capkun et al. 2008, McCune et al. 2005, Stajano and Anderson 1999, Goodrich et al. 2006, Roth et al. 2008, Mayrhofer and Gellersen 2007]. For example, devices can exchange keys over a visual channel between an LCD and a camera [McCune et al. 2005], an audio channel [Goodrich et al. 2006], an infrared channel [Balfanz et al. 2004], a dedicated wireless channel allocated exclusively for key exchange [Capkun et al. 2008], etc. Given the cost, size, and capability constraints imposed on many WiFi products, it is difficult for the industry to adopt a solution that requires an out-of-band communication channel.

This book presents *tamper-evident pairing* (TEP), a novel protocol that provides simple, secure WiFi pairing and protects against MITM attacks without an out-of-band channel. TEP can also be incorporated into PBC devices and existing WiFi chipsets without hardware changes.

TEP's main challenge in avoiding MITM attacks comes from operating on a shared wireless network, where an adversary can mask an attack behind cross-traffic, making it difficult to distinguish an adversary's actions from legitimate traffic patterns. To understand this, consider a key exchange between Alice and Bob, where Bob sends his Diffie–Hellman public key to Alice. Lucifer, the adversary, could tamper with this key exchange as follows.

Collision. Lucifer can jam Bob's message, causing a collision, which would not look out-of-the-ordinary on a busy wireless network. The collision prevents Alice from decoding Bob's message. Lucifer can now send his own message to Alice, in lieu of Bob's message, perhaps with the help of a directional antenna so that Bob does not notice the attack.

Capture effect. Lucifer can transmit simultaneously with Bob, but at a significantly higher power, to produce a capture effect at Alice [Ware et al. 2000]. In this case, Alice will decode Lucifer's message, in which he impersonates Bob, despite Bob's concurrent transmission. Bob will not know about Lucifer's transmission.

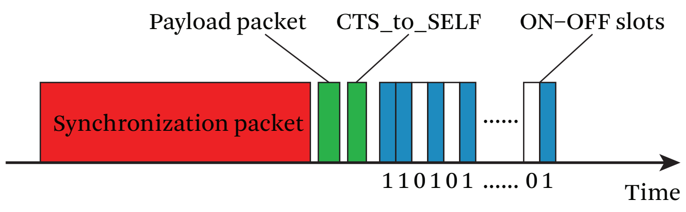


Figure 5.1 **The format of a tamper-evident message (TEM).**

Timing control. Lucifer can try to impersonate Alice by continuously occupying the wireless medium after Bob sends out his key, so that Lucifer can send out a message pretending to be Alice, but Alice does not get a chance to send her legitimate key.

To address these attacks in TEP, we introduce a *tamper-evident message* (TEM) primitive. The key characteristics of a TEM message is that an attacker can neither hide a TEM transmission from other nodes within radio range nor modify the content of the TEM without being detected. Thus, a TEM provides stronger guarantees than payload integrity because it also protects the fact that a message was transmitted in the first place.

Figure 5.1 shows the structure of a TEM. First, to ensure that Lucifer cannot mask Bob's TEM message by introducing a collision, the TEM starts with an exceptionally long packet. Since standard WiFi collisions are significantly shorter, Alice needs to detect only exceptionally long collisions (i.e., exceptionally long bursts of energy) as potential attacks on the key exchange process.

Second, to ensure that Lucifer cannot alter the payload of Bob's TEM by transmitting his own message at a high power to create a capture effect, we force any TEM message to include silence periods. As shown in Figure 5.1, the payload of the TEM message is followed by a sequence of short equal-size packets, called *slots*, where the transmission of a packet is interpreted as a "1" bit, and an idle medium is interpreted as a "0" bit. The bit sequence produced by the slots must match a hash of the TEM payload. If Lucifer overwrites Bob's message with his own, he must transmit slots corresponding to a hash of his message, including staying silent during any zero hash bits. However, since the hash of Lucifer's message differs from that of Bob's message, Bob's message will show up on the medium during Lucifer's "0" slots. Alice will detect a mismatch between the slots and the message hash and reject Lucifer's message.

Third, to ensure that legitimate nodes do not mess up the timing of Alice and Bob's key exchange, the TEM message includes a CTS-to-SELF, as shown in Figure 5.1. CTS-to-SELF is an 802.11 message that requires honest nodes to refrain from transmitting for a time period specified in the packet. TEP leverages this message for two goals. First, it uses it to reserve the medium for the duration of the TEM slots to ensure that legacy 802.11 nodes, unaware of the structure of a TEM message, do not sense the medium as idle and transmit during a TEM's silent slots. Second, TEP also uses CTS-to-SELF to reserve the medium for a short period after the TEM slots, to enable Alice to send her key to Bob within the interval allowed by PBC. Once Alice starts her transmission, the medium will be occupied, and honest 802.11 nodes will abstain from transmitting concurrently. If Lucifer transmits during the reserved time frame, Alice will still transmit her TEM message, and cause a collision, and hence an invalid TEM message that Bob can detect.

We build on TEM to develop the TEP pairing protocol. TEP exploits the fact that any attempts to alter or hide a TEM can be detected. Thus, given a pairing window, any attempt by an adversary to interfere with the pairing exchange translates into either an increase in the number of TEM messages or some invalid TEM messages. This allows the pairing devices to detect the attack and indicate to the user that pairing has failed and that she should retry. The cost of such a mechanism is that the user has to wait for a pre-determined duration of the pairing window. In Section 5.4.4, we describe how one may eliminate this wait by having a user push the button on a device a second time.

This book formalizes the above ideas to address possible interactions between the pairing devices, adversaries, and other users of the medium, and formally proves that the resulting protocol is secure against MITM attacks. Further, we build a prototype of TEP as an extension to the Ath5k wireless driver, and evaluate it using off-the-shelf 802.11 Atheros chipsets. Our findings are as follows.

- TEP can be accurately realized using existing OS and 802.11 hardware. Specifically, our prototype sender can schedule silent and occupied slots at a resolution of 40 μs, and its 95th percentile scheduling error is as low as 1.65 μs. Our prototype receiver can sense the medium's occupancy over periods as small as 20 μs and can distinguish occupied slots ("1" bits) from silent slots ("0" bits) with a zero error rate.
- Results from running the protocol on our campus network and applying the traces from the network during the SIGCOMM 2010 conference show that TEP never confuses honest 802.11 traffic for an attack. Furthermore, although our implementation is for 802.11, it can coexist with nearby Bluetooth devices which

do not respect TEP silent slots. In this case, TEP can still exchange a key using 1.4 attempts, on average.

Contributions. This book presents, to our knowledge, the first wireless pairing protocol that defeats MITM attacks without any key distribution or out-of-band channels. It does so by introducing TEM, a new key exchange message constructed in a manner that ensures an adversary can neither hide the fact that a message was transmitted, nor alter its payload without being detected. Our protocol is prototyped using off-the-shelf 802.11 devices and evaluated in production WiFi networks.

5.1 Related Work

There has been a lot of interest in user-friendly secure wireless pairing, which has led to a number of innovative solutions [Capkun et al. 2008, McCune et al. 2005, Balfanz et al. 2004, Stajano and Anderson 1999, Goodrich et al. 2006, Roth et al. 2008, Mayrhofer and Gellersen 2007]. TEP builds on this foundational work. However, TEP is the first to provide a secure pairing scheme that defeats MITM attacks without out-of-band channels, or key distribution or verification.

Closest to TEP is the work on integrity codes [Cagalj et al. 2006], which protects the integrity of a message's payload by inserting a particular pattern of ON–OFF slots. Integrity codes, however, assume a dedicated out-of-band wireless channel. In contrast, on shared channels, honest nodes may disturb the ON–OFF pattern by acquiring the medium during the OFF slots. Further, the attacker can hide the fact that a message was transmitted altogether, by using collisions or a capture effect. We build on integrity codes, but introduce TEM, a new communication primitive that not only protects payload integrity but also ensures that an attacker cannot hide that a message was transmitted. We further construct TEP by integrating TEM with the 802.11 standard, the PBC protocol, and the existing OS network stack. Finally, we implement TEP on off-the-shelf WiFi devices and evaluate it in operational networks.

TEP is also related to work on secure pairing, which traditionally required the user to either enter passwords or PINs [Boyko et al. 2000, IEEE 2002], or distribute public keys (e.g., STS [Diffie et al. 1992], Radius in 802.11i [IEEE 2004], or any other public key infrastructure). These solutions are appropriate for enterprise networks and for a certain class of home users who are comfortable with security setup. However, the need to ease security setup for non-technical home users has motivated multiple researchers to propose alternative solutions for secure pairing. Most previous solutions use a trusted out-of-band communication channel for key exchange. The simplest channel is a physical wired connection between the two devices. Other variants of out-of-band channels include the use of a display and a camera [McCune et al. 2005], an audio-based channel [Goodrich et al. 2006], an infra-red channel [Balfanz et al. 2004], a tactile channel [Stajano and Anderson 1999], or an accelerometer-based channel [Mayrhofer and Gellersen 2007]. While these proposals protect against MITM attacks, many devices cannot incorporate such channels due to size, power, or cost limitations. In contrast, TEP eases the security setup for home users and defeats MITM attacks, without any out-of-band channel.

Finally, multiple user studies [WiFi Alliance 2006, Kelton Research 2006, Norman 2009] have emphasized the difficulty in pairing devices for ordinary users. Our work is motivated by these studies. TEP requires the user to just push a button on each device—exactly as in PBC—and does not require any additional user involvement in key generation or verification.

5.2 PBC and 802.11 Background

5.2.1 Push Button Configuration

The WiFi-Alliance introduced the Push Button Configuration (PBC) mechanism to ease the security setup process for ordinary users, and to deal with devices that do not have an interface to enter passwords or PINs. In this section, we provide an overview of how PBC works.

Consider a home user who wants to associate an *enrollee* (PBC's term for the new device, e.g., a gaming console) with a *registrar* (PBC's term for, effectively, the access point). The user first pushes a button on the enrollee and then, within 120 s (called the walk time), pushes the button on the registrar. Once the buttons are pushed on the two devices, the devices perform a Diffie–Hellman key exchange to establish a secret key.

As shown in Figure 5.2, once the button is pushed on the enrollee, it periodically sends probes [WiFi Alliance 2006] requesting replies from registrars whose PBC button has been pressed. Once the enrollee receives a reply, it makes a note of the reply and continues to scan all the 802.11 channels for additional replies. If the enrollee receives replies from more than one registrar, across all 802.11 channels, it raises a session overlap error, indicating that the user should try again later. On the other hand, if it receives a reply from only one registrar, it proceeds with the registration protocol, using the Diffie–Hellman key from that one reply.

A registrar, for its part, stays on its dedicated channel and replies to probe requests only if the user has pushed its PBC button. Once the button is pushed, the registrar replies to PBC requests from potential enrollees. To detect conflicts, the registrar checks for requests in the last 120 s. If there are requests from more than one enrollee,

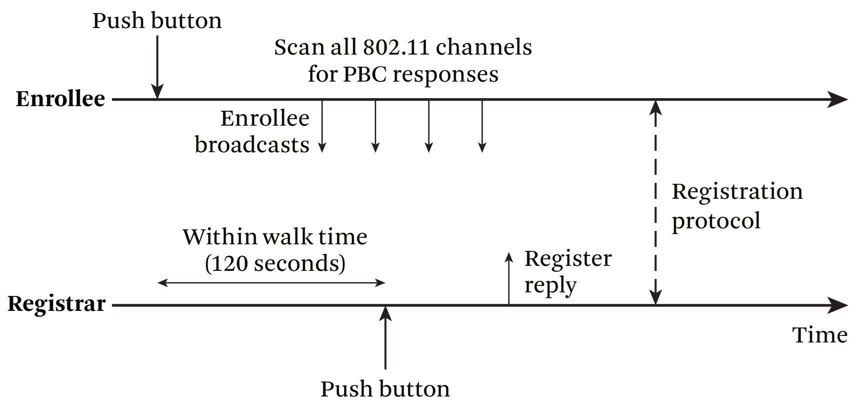


Figure 5.2 **A timeline depicting the operation of Push Button Configuration (PBC) between an enrollee and a registrar.**

the registrar signals a session overlap error and refuses to perform the PBC registration protocol, requiring the user to retry. If there was only one enrollee request, the registrar proceeds with the registration protocol using the Diffie–Hellman public key from that one request.

While PBC's use of Diffie–Hellman protects the devices from eavesdropper attacks, an active adversary can hide or change any of the messages, by resorting to collisions, capture effect attacks, or hogging the medium and delaying these messages. This allows an adversary to gain access to the user's registrar (e.g., their home network) or the enrollee device, or to intercept and alter any future messages between the enrollee and registrar. Defending against such adversaries requires a system that is robust to MITM attacks, which is the main contribution of TEP.

5.2.2 802.11

Since our protocol involves low-level details of the 802.11 standard, we summarize the relevant aspects of 802.11 in this section. The 802.11 standard requires nodes to sense the wireless medium for energy, and transmit only in its absence. 802.11 nodes can transmit using a range of bit rates, with the minimum bit rate of 1 Mbps. Coupling this with the fact that the maximum packet size used by higher layers is typically 1500 bytes, an honest node can occupy the channel for a maximum of 12 ms. 802.11 requires back-to-back packets to be separated by an interval called the DCF Inter-Frame Spacing (DIFS), whose value can be 34 μs, 50 μs, or 28 μs, depending on whether the network uses 802.11a, b, or g. 802.11 acknowledgment packets, however, can be transmitted after a shorter duration of 10 μs, called the Short Inter-Frame Spacing (SIFS).

5.3 Security Model

TEP addresses the problem of authenticating key exchange messages between two wireless devices, in the presence of an active adversary that may try to mount a man-in-the-middle attack.

5.3.1 Threat Model

The adversary can eavesdrop on all the signals on the channel, including all prior communications. The adversary can also be active and transmit with an arbitrary power, at any time, thereby corrupting or overpowering other concurrent transmissions. The adversary may know the TEP protocol, the precise times when devices transmit their announcements, and their exact locations. In addition, the adversary can know the exact channel between the pairing devices and the channel from the pairing devices to the adversary. The adversary can also be anywhere in the network and is free to move. Multiple adversaries may exist in the network and can collude with each other.

The adversary can have access to state-of-the-art RF technologies: he can have a multi-antenna system, he may be able to simultaneously receive and transmit signals, and he can use directional antennas to ensure that only one of the pairing devices can hear its transmissions.

The adversary, however, does *not* have physical control over the pairing devices or their surroundings. Specifically, the adversary cannot place either of the two devices in a Faraday cage to shield all signals. We also assume that the adversary cannot break traditional cryptographic constructs, such as collision-resistant hash functions.

Finally, we assume that the PBC buttons operate according to the PBC standard [WiFi Alliance 2006] and that the user performs the PBC pairing as prescribed in the standard, i.e., the user puts the two devices in range then pushes the buttons on the two devices within 120 s of each other.

5.3.2 Security Guarantees

Under the assumptions outlined above, TEM guarantees that an adversary cannot tamper with the payload of a TEM message or mask the fact that a TEM message was transmitted. Building on the TEM mechanism, TEP guarantees that in the absence of an active adversary, two pairing devices can establish secure pairing. In the presence of an adversary who is actively mounting MITM attacks (or in the presence of more than two devices attempting to pair at the same time), TEP ensures that the pairing devices will signal an error and never be tricked into pairing with the adversary (or, more generally, with the wrong device). In other words, TEP provides the PBC security guarantees augmented with protection against MITM attacks.

5.4 TEP Design

TEP's design is based upon the TEM mechanism, a unidirectional announcement protocol that guarantees that adversaries cannot tamper with or mask TEM messages without detection. TEP uses TEM to exchange public keys between the PBC enrollee and registrar in a way that resists MITM attacks. At a high level, when an enrollee enters PBC mode, it sends out a TEM message containing its public key. When a registrar in PBC mode receives this message (or suspects that an adversary may have tried to tamper with or mask such a message), it responds with its own public key. Both the enrollee and registrar collect all TEM messages received during PBC's walk time period. If, during that time, each received exactly one unique public key (and no tampered messages), they can conclude that this public key came from the other party, and can use it for pairing. Otherwise, PBC reports a session overlap error (e.g., because multiple enrollees or registrars were pairing at the same time, or because an adversary interfered), and asks the user to retry.

The rest of this section describes our protocol in more detail, starting with the TEM mechanism, using terminology defined in Table 5.1.

5.4.1 Tamper-Evident Message (TEM)

The goal of TEM is to guarantee that if an attacker tampers with the payload of a TEM message, or tries to mask the fact that a message was transmitted at all, a TEM receiver within communication range will detect such tampering. In other words, TEM receivers will *always* detect when a TEM message was, or may have been, transmitted.

To provide this guarantee, TEM messages have a specialized structure, as shown in Figure 5.1. First, there is a synchronization packet, which protects the TEM's transmission from being masked, by unambiguously indicating to a TEM receiver that a TEM message follows. The synchronization packet contains random data, to ensure that an adversary cannot cancel out its energy.[3]

Second, the TEM message contains the announcement payload. The payload is always of fixed length, to ensure that an adversary cannot truncate or extend the payload in flight, but otherwise has no restrictions on its content or encoding. In our pairing protocol, the payload of a TEM message contains the sender's Diffie–Hellman public key, along with other registration information.

3. In practice, it is very hard to cancel a signal in flight, but in theory an attacker who knows the transmitted signal and the channels to the receiver can construct a signal that cancels out the original signal at the receiver. Making the data random eliminates this option.

Table 5.1 **Terminology used to describe TEP**

Term	Definition
Tamper-evident message	A wireless message whose presence and the integrity of its payload are guaranteed to be detected by every receiver within radio range (Figure 5.1).
Synchronization packet	An exceptionally long packet whose presence indicates a TEM. To detect a synchronization packet, it is sufficient to detect that the medium is continuously occupied for the duration of the synchronization packet, which is 19 ms.
Payload packet	The part of a TEM containing the data payload (e.g., a device public key).
ON–OFF slot	The interval used to convey one bit from sender to receiver. The slot time is 40 μs. The bits in the slots are balanced, as described in Section 5.4.1.
Occupied/ON slot	A slot during which the medium is busy with a transmission.
Silent/OFF slot	A slot during which the medium is idle.
Sensing window	The interval over which the receiver collects aggregate information for whether the medium is occupied or silent.
Fractional occupancy	The fraction of time the medium was busy during a sensing window.

Third, the TEM message contains ON–OFF slots, which guarantee that any tampering with a TEM payload is detectable. Similar to the synchronization packet, the content of the ON slots is randomized. The first two slots, as shown in Figure 5.3, encode the direction flag, which defines whether this TEM message was sent by an enrollee (called a TEM request, flag value “10”) or by a registrar (called a TEM reply, flag value “01”). The remaining slots contain a cryptographic hash of the payload. While it is possible to also encode the payload using slots, it would be inefficient for long payloads, and unnecessary, since protecting a cryptographic hash suffices. To detect tampering, TEM encodes all slots in a way that guarantees that exactly half of the slots are silent, as we describe in Section 5.4.1.

(a) Detecting Tampering. To determine if an adversary may have tampered with a TEM message, a TEM receiver performs several checks. First, the receiver continuously monitors the medium for possible synchronization packets. If it detects any burst of energy

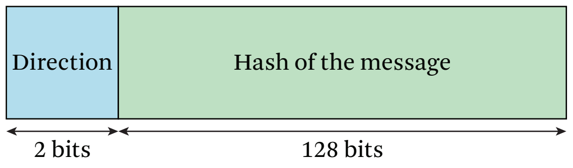


Figure 5.3 **Data encoded in the ON–OFF slots.** The first two bits specify the direction of the message, and the rest of the bits contain a cryptographic hash of the payload.

at least as long as the synchronization packet, it interprets it as the start of a TEM announcement. The receiver conservatively assumes that any such period of energy is a TEM message, and signals a missed message if it is unable to decode and verify the subsequent payload. To minimize false positives, we choose a synchronization packet that is longer than any regular contiguous WiFi transmission. An adversary cannot cancel out a legitimate synchronization packet, because the adversary cannot eliminate the power on the channel. In fact, since the payload of the synchronization packet is random, the adversary cannot cancel the power from the packet even if he knows the exact channel between Alice and Bob and is fully synchronized with the transmitter. Thus, an adversary cannot tamper with the presence of a TEM message by masking it out.

Second, once a TEM receiver detects the start of a TEM announcement, it attempts to decode the payload packet and the hash bits in the ON–OFF slots. If the receiver cannot decode the payload (i.e., the packet checksum fails), it indicates tampering. If the payload is decoded, the receiver verifies that the hash bits match the hash of the payload, i.e., it verifies that hashing the payload produces the same bits in the ON–OFF slots and that the number of ON slots is equal to that of OFF slots. If the receiver cannot verify the hash bits, it conservatively assumes that an adversary is tampering with the transmission. Once tampering is detected, the receiver signals a session overlap error (as in PBC), requiring the user to retry later.

(b) Balancing the ON–OFF Slots. An adversary can transform an OFF slot to an ON slot (by transmitting in it) but cannot transform an ON slot to an OFF slot. Hence, to ensure that the adversary cannot tamper with even a single OFF slot without being detected, we make the number of the OFF slots in a TEM message equal to that of the ON slots, i.e., we balance the slots. The number of slots is fixed by the TEP protocol, thus avoiding truncation or extension attacks. Since the direction flag is already encoded in two balanced bits, we now focus on balancing the rest of the slots.

Our balancing algorithm takes the hash bits of the TEM payload and produces a balanced bit sequence to be sent in the ON–OFF slots. One inefficient but simple transformation is to use Manchester encoding of the hash bits to produce a balanced output bit sequence with twice as many output bits. TEM, however, introduces an efficient encoding that takes an even number, N, of input bits and produces $M = N + 2\lceil \log N \rceil$ output bits which have an equal number of zeros and ones. The details of our efficient encoding algorithm are presented in [Gollakota et al. 2011].

(c) Interoperating with 802.11. To interoperate with other 802.11 devices that may not be TEM-aware, the ON–OFF slots are preceded by a CTS-to-SELF packet, which reserves the medium for the TEM message. This serves two purposes. First, since the sender does not transmit during the OFF slots, another 802.11 node could sense the wireless medium to be idle for more than a DIFS period and start transmitting its own packet during that OFF slot. The 802.11 standard requires 802.11 nodes that hear a CTS-to-SELF on the channel to abstain from transmitting for the period mentioned in that packet, which will ensure that no legitimate transmission overlaps with the slots. Second, in case of a TEM message from an enrollee to a potential registrar, the CTS-to-SELF packet reserves the medium so that the registrar can immediately reply with its own TEM message. This prevents legitimate nodes from hogging the medium and delaying the registrar's response. However, reserving the channel for the entire length of a TEM message is inefficient if no registrar is present. To avoid under-utilization of the wireless medium, the enrollee's CTS-to-SELF only reserves the channel for a DIFS period past its slot transmissions. If a PBC-activated registrar is present, it *must* start transmitting its response message within the DIFS period. On the other hand, if there is no registrar, other legitimate devices will resume transmissions promptly.

To maximize the probability that all devices can decode the CTS-to-SELF, it is transmitted at the most robust bit rate of 1 Mbps. Current 802.11 implementations obey a CTS-to-SELF that reserves the channel up to 32 ms. Our TEM message requires 144 slots,[4] and the slot duration is 40 μs (Section 5.5). This translates to about 5.8 ms, which is less than the 32 ms allowed by the CTS-to-SELF.

Finally, as shown in Figure 5.1, there is a gap between the synchronization and payload packets. If this gap is large, other 802.11 nodes would sense an idle wireless medium and start transmitting, thus appearing to tamper with the TEM. To avoid

4. Two of the slots are for the direction bit, and the remaining 142 are for the bit-balanced hash bits. More specifically, the bit-balancing algorithm, in Section 5.4.1, takes N input bits and outputs $N + 2\lceil \log N \rceil$ bits. Since the hash is a 128 bit function, the bit-balancing algorithm produces 142 bit-balanced hash bits.

this, we exploit the fact that 802.11 nodes are only allowed to transmit if they find the medium continuously idle for a DIFS. Thus, a TEM sender sends the payload packet immediately after the synchronization packet with a gap of a SIFS, which is much less than a DIFS.

(d) API Summary. For the sender side, TEM provides a single blocking function,

- void TEM_SEND (bool *dir*, str *msg*, time *t*),

which sends an announcement containing payload *msg*. The *dir* flag specifies the direction of the message; i.e., whether it is a request message (from the enrollee) or a reply message (from the registrar). Time *t* specifies the deadline by which the message must start transmission. The TEM sender tries to respect carrier-sense in the medium access control (MAC) protocol, and waits until the medium is idle before transmitting its message. However, if the message cannot be transmitted by time *t* (e.g., because an adversary is hogging the medium), the sender overrides the MAC's carrier-sense, and transmits the announcement anyway, so that recipients will detect tampering. Note that the CTS-to-SELF requires honest nodes to release the medium for the registrar to transmit its own TEM reply.

For the receiver side, TEM provides two functions:

- handle TEM_RECV_START (bool *dir*), and
- msg_list TEM_RECV_GET (handle *h*).

The first function, TEM_RECV_START, starts listening on the wireless medium for TEM messages that are either requests (from an enrollee) or replies (from a registrar), based on the *dir* flag. The second function, TEM_RECV_GET, is used to retrieve the set of messages accumulated by the receiver since TEM_RECV_START or TEM_RECV_GET was last invoked. If TEM_RECV_GET could not decode a possible TEM message (or verify that it was not tampered with), it returns a special value RETRY, which causes the caller (i.e., TEP) to re-run its protocol. As an optimization, if *all* of the TEM messages that TEM_RECV_GET was unable to decode were overlapping with the receiver's own transmissions (i.e., a concurrent TEM_SEND), TEM_RECV_GET returns a special value OVERLAP instead of RETRY. We describe in Section 5.5.4 how a node detects TEM messages that overlap with its own transmissions, and in [Gollakota et al. 2011] how we use the overlap information to optimize wireless medium utilization.

5.4.2 Securing PBC using TEM

Using the TEM mechanism, we will now describe how TEP—a modified version of the PBC protocol—avoids man-in-the-middle attacks.

Once the button is pressed on the enrollee, the enrollee repeatedly scans the 802.11 channels in a round-robin manner, as in the current PBC protocol. On each channel, the enrollee transmits a TEM request, i.e., a TEM message with the direction flag set to "10". The TEM request contains the enrollee's public key (and any PBC information included in an enrollee's probe). If an adversary continuously occupies the medium for *tx_tmo* (e.g., 1 s), the enrollee overrides carrier-sense and transmits its message anyway. The enrollee then waits for a TEM response from a registrar, which is required to immediately respond. The enrollee records the responses, if any, and after a specified period on each channel it moves to the next 802.11 channel and repeats the process. The enrollee continues to cycle through all 802.11 channels for PBC's walk time period. The enrollee's logic corresponds to the following pseudo-code to build up r, the set of registrar responses:

```
$r \leftarrow \varnothing$
for 120 sec $+ \#channels \times (tx\_tmo + 2 \times tem\_duration)$ do      ▷ walk time + max
                                                                            ▷ enrollee scan period
    switch to next 802.11 channel
    $h \leftarrow$ TEM_RECV_START (reply)
    TEM_SEND (request, $enroll\_info$, $now + tx\_tmo$)
    SLEEP ($tem\_duration$)
    $r \leftarrow r \cup$ TEM_RECV_GET($h$)
end for
```

A registrar follows a similar protocol. Once the PBC button is pressed, the registrar starts listening for possible TEM requests on its 802.11 channel. Every time a TEM message is received, the registrar records the message payload, and immediately sends its own TEM message in response, containing the registrar's public key. It is safe to reply immediately because the sender's TEM message ended with a CTS-to-SELF, which reserved the medium for the registrar's reply. The registrar's pseudo-code to build up e, the set of enrollee messages, is as follows:

```
$e \leftarrow \varnothing$
$h \leftarrow$ TEM_RECV_START (request)
for 120 sec $+ \#channels \times (tx\_tmo + 2 \times tem\_duration)$ do      ▷ walk time + max
                                                                            ▷ enrollee scan period
    $m \leftarrow$ TEM_RECV_GET ($h$)
    if $m \neq \varnothing$                                                  ▷ enrollee, RETRY, or OVERLAP
        $e \leftarrow e \cup m$
        TEM_SEND (reply, $registrar\_info$, $now$)                            ▷ send reply immediately
    end if
end for
```

After the PBC's walk time expires, both the enrollee and the registrar check the list of received messages. Successful pairing requires that both the enrollee and the registrar receive exactly one unique public key via TEM messages, and that no messages were tampered with (i.e., TEM_RECV_GET never returned RETRY or OVERLAP). If exactly one public key was received, it must have been the public key of the other party, and TEP can safely proceed with pairing. If more than one public key was received, or RETRY or OVERLAP was returned, then a session overlap error is raised, indicating that more than one pair of devices may be attempting to pair, or that an adversary is mounting an attack. In this situation, the user must retry pairing.

(a) Reducing Medium Occupancy. The protocol described above is correct and secure (as we will prove in Section 5.6.1). However, it can be inefficient if somehow multiple registrars transmit overlapping replies at almost the same time. Each of them will then assume it may have missed a request from some enrollee (since it sensed a concurrent TEM message), and each will re-send its reply. This cycle may continue for the walk time of 120 s, unnecessarily occupying the wireless medium. Gollakota et al. [2011] describes an optimization that avoids this situation and we prove that the optimized protocol maintains the same security guarantees.

5.4.3 Example scenarios

Figure 5.4 shows how TEP works in five potential scenarios. In scenario (a), there is no attacker. In this case, the enrollee sends a request to which the registrar replies immediately. The two devices can thus proceed to complete pairing after 120 s. In scenario (b), the enrollee transmits its request, but the attacker immediately jams it so that the registrar can not decode the enrollee's request. However, the registrar detects a long burst of energy, which the registrar interprets as a TEM announcement, causing it to reply to the enrollee.

In scenario (c), the enrollee sends the request; the attacker then captures the medium at the same time as the registrar, and transmits a reply, at a high power, impersonating the registrar. Because of capture effect, the enrollee decodes the message payload from the attacker. But since the registrar and the attacker transmit the hash function of different messages in the ON–OFF slots, the enrollee notes that the slots do not have equal number of zeros and ones and hence detects tampering with the announcement.

In scenario (d), the adversary sends a request message in an attempt to gain access to the registrar; as stipulated by TEP, the registrar replies to this request. However, since the registrar waits for 120 s before completing the pairing, it also hears the request from the enrollee. Since the registrar receives requests from two devices, it raises a session overlap error.

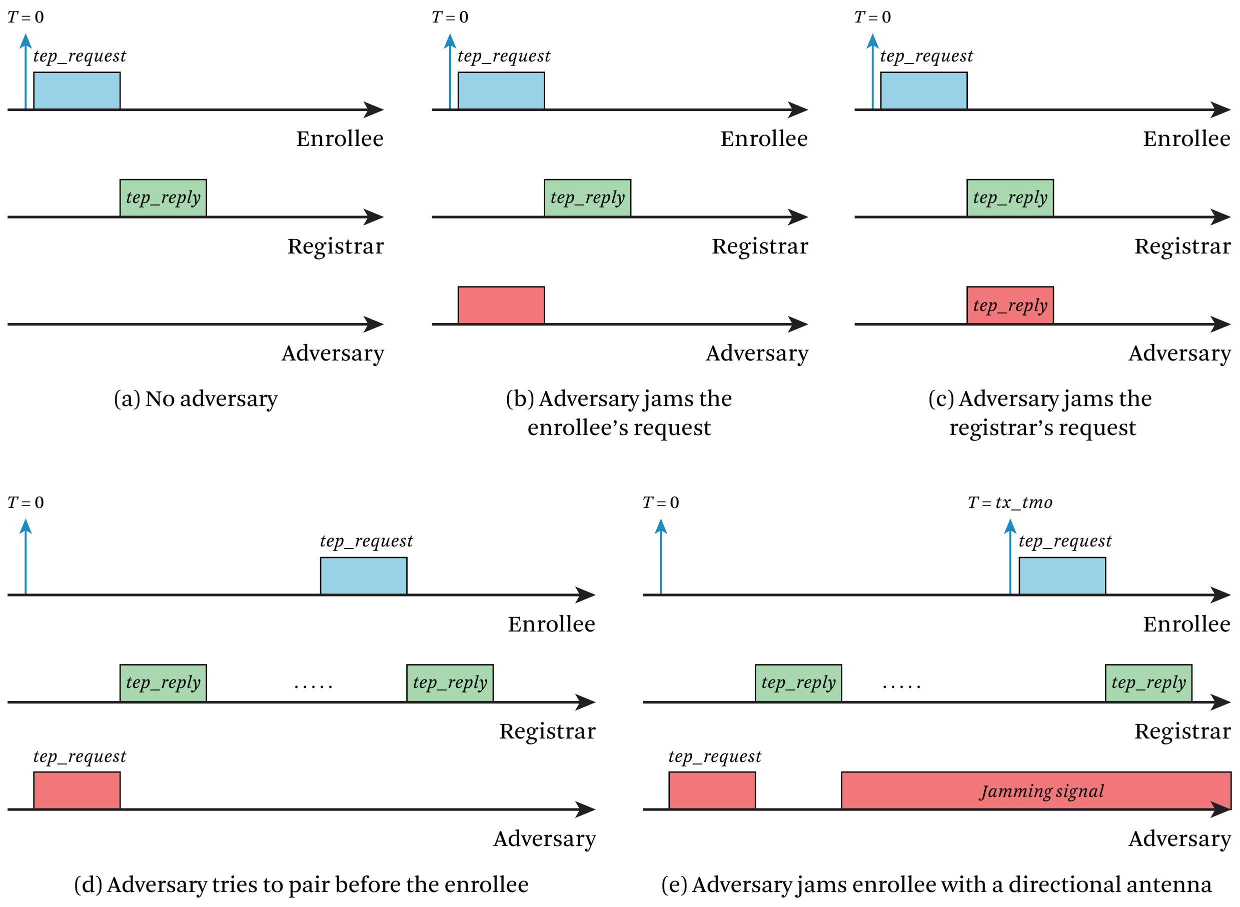

T = 0
tep_request
Enrollee
tep_reply
Registrar
Adversary
(a) No adversary
(b) Adversary jams the enrollee's request
tep_reply
(c) Adversary jams the registrar's request
.....
T = tx_tmo
Jamming signal
(d) Adversary tries to pair before the enrollee
(e) Adversary jams enrollee with a directional antenna

Figure 5.4 **Timelines of five example runs of the TEP protocol.**

Finally, in scenario (e), the adversary sends a TEM request, receives the registrar's reply, and then continuously jams the enrollee using a directional antenna. By using a directional antenna, the adversary ensures that the registrar does not detect the jamming signal and hence does not interpret it as an invalid TEM. The enrollee carrier-senses, detects that the medium is occupied, and does not transmit until it times out after tx_tmo seconds, at which point it ignores carrier sense and transmits its TEM request. The registrar listens to this request message and detects the presence of the enrollee. Since the registrar receives requests from two devices, it raises a session overlap error.

5.4.4 Making Pairing Faster

The extension of PBC to use TEM, described above, requires the enrollee and registrar to wait for 120 s before completing the association process. If the enrollee does not wait for a full 120 s, and simply picks the first responding registrar, it may pick an adversary's registrar—a legitimate registrar only replies when its PBC button has been pushed, and the user might push the registrar's PBC button slightly later than the enrollee's. Because the enrollee does not know if the user has already pushed the registrar's button, it has to wait for 120 s to be sure that the user has pushed the button. In this section, we describe how one can eliminate this delay.

First, if the user always pushes the enrollee's button before the registrar's button, then the registrar does not need to wait for 120 s; the registrar needs to wait for just the time it takes an enrollee to cycle through all of 802.11's channels (which is less than 12 s). Second, we can also eliminate the enrollee's wait time. Specifically, if the user explicitly tells the enrollee that the registrar's button was pushed, the enrollee can complete the association process after one cycle through the 802.11 channels, eliminating the additional wait time.

For example, one approach would be to have the user first press the button on the enrollee, then press the button on the registrar, and then again push the button on the enrollee. Note that, in this approach, the registrar does not have to wait for 120 s: because the registrar's button is always pushed after the enrollee, the registrar knows that the enrollee is active, and is guaranteed to see the enrollee's TEM message within the time required for the enrollee to cycle through all 802.11 channels. (Of course, if the 120 s period expires on the enrollee without any additional button pushes, the enrollee can proceed to completion as before, with two total button pushes from the user.)

5.5 TEM on Off-the-shelf Hardware

We implement TEM on Atheros AR5001X+ chipsets by modifying the ath5k driver and running TEM's timing-sensitive code in a kernel driver.

5.5.1 Scheduling Slot Transmission

To reduce the air time of a TEM, we must minimize the size of a single slot packet in the ON–OFF slots. Since the slot packet's payload need not be decoded (just the presence or absence of a slot packet conveys a 1 or 0 bit), we transmit slot packets at the highest bitrate, 54 Mbps, for a total of 40 μs.

In addition to reducing the size of a slot packet, TEM must transmit slot packets at precise slot boundaries. Queueing in the kernel and carrier-sense in the card make precise transmission timing challenging. We avoid kernel queueing by implementing

TEM in a kernel driver and using high-resolution timers. We avoid delays in the wireless card itself through several changes to the card firmware and driver, as follows. For the duration of the slots, we disable binary exponential backoff (802.11 BEB) by setting CW_{MIN} and CW_{MAX} to 1. To prevent carrier-sense backoff, we disable automatic noise calibration by setting the noise floor register to "high". We place slot packets in the high-priority queue. Finally, we disable the transmitter's own beacons by disabling the beacon queue. In aggregate, these changes allow us to make slot packets as short as 40 μs and maintain accurate slot timing.

5.5.2 Energy Detection at the Receiver

A TEM receiver detects a synchronization packet and distinguishes ON from OFF slots by checking the energy level on the medium. Hence, the receiver needs to distinguish the noise level, which is around -90 dB, from an actual transmission. To do this, we set the noise floor to -90 dB and deactivate auto-calibration while running TEP.[5]

While an ideal receiver would detect energy at the finest resolution (i.e., every signal sample), existing wireless chipsets do not give access to these samples. Instead, we exploit two registers provided by the ath5k firmware: *AR5K_PROFCNT_CYCLE* and *AR5K_PROFCNT_RXCLR*. The first register is incremented every clock cycle based on the clock on the wireless hardware. The second register, on the other hand, is increment only if the hardware finds high energy during that clock cycle.

Using these registers, we define a *sensing window (SW)* as the interval over which the receiver collects aggregate information for whether the medium is occupied or silent, as defined in Table 5.1. At the beginning of a SW, a TEM receiver resets both registers to 0, and reads them at the end of the SW. The ratio of these two registers at the end of the SW, $\frac{AR5K\_PROFCNT\_RXCLR}{AR5K\_PROFCNT\_CYCLE}$, is defined as the *fractional occupancy*. By putting a threshold on the fractional occupancy, a TEM receiver can detect whether the medium is occupied in a particular SW, and hence can detect energy bursts and measure their durations in units of the sensing window. Similar to the sender, a TEM receiver runs in the kernel to precisely schedule sensing windows.

Our implementation dynamically adjusts the length of the sensing window to minimize system overhead. The TEM receiver uses a long sensing window of 2 ms, until it

5. There is a tradeoff between the noise floor and the permissible distance between the pairing devices. In particular, pairing devices separated by large distances have a weak signal, and hence, to ensure detection, the noise floor should be set to a low value. On the other hand, pairing devices that are closer have a stronger signal, and hence the noise floor can be set to a higher value. We pick -90 dB because it is the default noise floor value in typical WiFi implementations. Manufacturers, however, can pick a higher default value, as long as the pairing devices are placed closer to each other.

detects a burst of energy longer than 17 ms. This indicates a synchronization packet, at which point the receiver switches to a 20 μs sensing window to accurately measure energy during slots, providing on average two sensing window measurements for every slot.

The receiver must be careful to ensure that a 20 μs sensing window allows accurate detection of slot occupancy. But, because the sender and receiver are not synchronized, sensing windows may not be aligned with slots, and in the worst case, will be off by half a sensing window, i.e., 10 μs. However, having a sensing window that is half the length of a slot ensures that at least one of every two sensing windows is completely within a slot (i.e., does not cross a slot boundary). Thus, to measure slot occupancy, the receiver compares the variance of odd-numbered sensing window measurements and even-numbered sensing window measurements, and uses the one with the highest variance. Because the slots are bit-balanced, the correct sequence will have an equal number of ones and zeros, having the higher variance.

This technique for measuring slot occupancy is secure in the presence of an adversary. As we will prove in Proposition 5.1, an adversary can introduce energy, but cannot cancel energy in an occupied slot. Thus, the adversary can only increase—but cannot reduce—the computed occupancy ratios in either the odd or the even windows. As a result, the adversary cannot create a different bit sequence in either the odd or even windows which still has an equal number of ones and zeros. Thus, sampling at twice the slot rate maintains TEM's security guarantees.

5.5.3 Sending A Synchronization Packet

To transmit a long synchronization packet, TEM transmits the maximum-sized packet allowed by our hardware (2400 bytes) at the lowest bit rate (1 Mbps), resulting in a 19 ms synchronization packet. While many receivers drop such long packets (the maximum packet size permissible by the higher layers is 1500 bytes), this does not affect a TEM receiver, since it does not need to decode the packet; it only needs to detect a long burst of energy.

5.5.4 Checking for TEM While Transmitting

While executing the TEP protocol (which lasts for 120 s), a node must detect TEM messages transmitted by other nodes even if they overlap with its own transmissions. We distinguish two cases: First, when the node transmits a standard 802.11 packet, it conservatively assumes that the channel has been occupied by part of a synchronization packet for the duration of its transmission. The node samples the medium before and after its transmission, checking for continuous occupancy by a synchronization packet. As our evaluation shows (Section 5.6.3), the longest packets in operational

WiFi networks are about 4 ms (a collision of two packets sent at the lowest 802.11g rate of 6 Mb/s), making synchronization packet false positives unlikely even with the conservative assumption that the entire 4 ms transmission overlapped with part of a synchronization packet (19 ms).[6]

Second, a node that is transmitting a TEM request must not miss a concurrently transmitted TEM reply, and similarly a node that is transmitting a reply must not miss a concurrent request. To detect partially overlapping TEM messages, a node samples the medium before and after every synchronization packet, and after the slots of every TEM message, and if it detects energy, it assumes that it may have missed an overlapping TEM message (and thus, TEM_RECV_GET will return OVERLAP, unless it observes other possibly missed messages, in which case it will return RETRY.) Since the total length of the ON–OFF slots is shorter than the length of the synchronization packet, sampling the medium after the end of a synchronization packet (i.e., before the start of the payload and slots) and after the end of the slots suffices to detect an overlapping synchronization packet. Finally, in the case when two TEM messages are perfectly synchronized, the node uses the direction bits to detect a collision. Since the direction flag for a request is "10" and a reply "01", the node checks for this scenario by checking the energy level during the OFF slot in the direction field in its own transmission. If the OFF slot shows a high energy level, TEM_RECV_GET will return OVERLAP (or RETRY, if there are other missed messages).

5.6 Evaluation

We evaluate TEP along three axes: security, accuracy, and performance. Our findings are as follows.

- TEP is provably secure to MITM attacks.
- TEP can be accurately realized using existing OS and 802.11 hardware. Specifically, our prototype sender can schedule ON–OFF slots at a resolution of 40 μs, and its 95th percentile scheduling error is as low as 1.65 μs. Our prototype receiver can sense the medium's occupancy over periods as small as 20 μs and can distinguish ON slots from OFF slots with a zero error rate.
- Results from two operational networks—our campus network and SIGCOMM 2010—show that TEP never confuses cross-traffic for an attack. Further, even in the presence of Bluetooth devices which do not obey CTS-to-SELF and may trans-

6. Note that even if some networks have normal packets that are much larger than 4 ms, this may create false positives but does not affect the security of the protocol.

mit during TEP's OFF slots, TEP can perform key exchange in 1.4 attempts, on average.

5.6.1 Evaluating TEP's Security

We analyze TEP's security using the threat model in Section 5.3.1. To do so, we formally state our definitions, then prove that a TEM is tamper resistant and that wireless pairing using TEP is secure to MITM attacks.

Definition 5.1 Tamper-evident. A message is said to be tamper-evident if an adversary can neither change the message's content without being detected nor hide the fact that the message has been transmitted.

Before we proceed to prove that a TEM is tamper-evident, we first prove the following proposition about the capability of an adversary.

Proposition 5.1 Let $s(t)$ be the transmitted signal, and let $h(t)$ be the channel impulse function. Assuming the transmitted signal is unpredictable, and the receiver is within radio range of the sender, an adversary cannot cancel the signal energy at the receiver even if he knows the channel function between the sender and receiver, $h(t)$.

Proof The received signal is a convolution of the transmitted signal and the channel impulse function, plus the adversary's signal $a(t)$, plus white Gaussian noise $n(t)$, i.e., $r(t) = h(t) * s(t) + a(t) + n(t)$. To cancel the received energy, the adversary needs to produce a signal $a(t)$ so that $r(t) \approx n(t)$, or equivalently, $h(t) * s(t) + a(t) \ll n(t)$. Since the receiver is within radio range of the sender, we know $h(t) * s(t) \gg n(t)$, and, since $n(t)$ is physically unpredictable, we know that $a(t) \approx -h(t) * s(t)$. But an adversary that can compute such an $a(t)$ directly contradicts our assumption that $s(t)$ is unpredictable, and thus an adversary cannot compute such an $a(t)$. ■

Since the synchronization packet and ON slots have random contents, Proposition 5.1 implies that an adversary cannot hide the channel energy during the transmission of the synchronization packet or the ON slots from a receiver. Based on this result, we proceed to prove the following.

Proposition 5.2 Given the transmitter and receiver are within range, and the receiver is sensing the medium, a TEM, described in Section 5.4.1, is tamper-evident.

Proof We prove Proposition 5.2 by contradiction. Assume that one party, Alice, sends a TEM to a second party, Bob. Suppose that Alice's TEM to Bob fails to be tamper-evident. This can happen because the adversary succeeds either in hiding from Bob that Alice sent a TEM, or in changing the TEM content without being detected by Bob. To hide Alice's

TEM, the adversary must convince Bob that no synchronization packet was transmitted. This requires the adversary to cancel the energy of the synchronization packet at Bob, which contradicts Proposition 5.1. Thus, the adversary must have changed the announcement.

Suppose the adversary changed the data encoded in the slots. Proposition 5.1 says that the adversary cannot cancel the energy in an ON slot, and hence cannot change an ON slot to an OFF slot. Since the number of ON and OFF slots is balanced, the adversary cannot change the slots without increasing the number of ON slots and thus being detected. Thus, the only alternative is that the adversary must have changed the message packet. Since the ON–OFF slots include a cryptographic hash of the message, this means that the adversary constructed a different message packet with the same hash as the original message packet. This contradicts our assumption that the hash is collision resistant. Thus, the adversary cannot alter the announcement content, and TEM is tamper-evident. ■

Although Proposition 5.2 guarantees that a TEM message is tamper-evident if the receiver is sensing the medium, the receiver may be transmitting its own message at the same time. We now prove that a TEM is tamper-evident even if the receiver transmits.

Proposition 5.3 Given a receiver (Bob) that can send its own messages, a TEM sent by a transmitter (Alice) in range of the receiver is tamper-evident, if the receiver follows the concurrent-transmission protocol of Section 5.5.4 and the receiver and transmitter send TEM messages with different directions (request or reply).

Proof If Bob detects the synchronization packet (SP) of Alice's TEM, the TEM is tamper-evident: either Bob refrains from sending during that TEM, in which case Proposition 5.2 applies, or Bob transmits concurrently, and TEM_RECV_GET will return RETRY or OVERLAP.

If Bob fails to detect Alice's SP, it must have happened while Bob was sending his own message (otherwise, Proposition 5.2 applies). Since regular 802.11 packets are shorter than a SP, and Section 5.5.4 conservatively assumes the medium was occupied for the entire duration of the transmitted packet, Bob could not have missed a SP while sending a regular packet. Thus, the only remaining option is that Alice's SP overlapped with a TEM sent by Bob.

Consider four cases for when Alice's SP was sent in relation to the SP of Bob's TEM. First, if Alice's SP started before Bob's SP, Bob would detect energy before starting to transmit his SP and return OVERLAP or RETRY (Section 5.5.4), making the TEM tamper-evident. Second, if Alice's SP started exactly at the same time as Bob's SP, Bob would detect energy during the direction bits and return OVERLAP or RETRY (Section 5.5.4), making the TEM tamper-evident. Third, if Alice's SP started during Bob's SP, Bob would detect energy after his SP and return OVERLAP or RETRY (Section 5.5.4), making the TEM tamper-evident. Fourth, if Alice's SP started after Bob's SP ended, Bob would detect energy from Alice's SP after the end of his TEM slots and return OVERLAP or RETRY (Section 5.5.4), making the TEM tamper-evident. Thus, in all cases, the TEM is tamper-evident. ■

We now prove TEP is secure against a MITM attack.

Proposition 5.4 Suppose an enrollee and a registrar are within range, both are following the TEP protocol as described in Section 5.4.2, and the user does the stipulated actions required by PBC. Under the threat model defined in Section 5.3.1, an adversary cannot convince either the enrollee or the registrar to accept any public key that is not the legitimate public key of the other device.

Proof We prove Proposition 5.4 by contradiction, considering first the registrar, and then the enrollee. First, suppose an adversary convinces the registrar to accept a public key other than that of the enrollee. By Section 5.4.2, this means the registrar received exactly one public key (and, thus, did not receive the enrollee's key), and that TEM_RECV_GET never returned OVERLAP or RETRY. By assumption, the enrollee and registrar entered PBC mode within 120 s of each other, which means they were concurrently running their respective pseudo-code for at least $\#channels \times (tx\_tmo + 2 \times tem\_duration)$ seconds, and therefore the enrollee must have transmitted at least one TEM message on the registrar's channel while the registrar was listening. Proposition 5.3 guarantees that the registrar must have either received that one message or detected tampering (and returned OVERLAP or RETRY), which contradicts our assumption that the registrar never received the enrollee's message and never returned OVERLAP or RETRY. Thus, an adversary cannot convince the registrar to accept a public key other than that of the enrollee.

Second, suppose an adversary convinces the enrollee to accept a public key other than that of the registrar. By Section 5.4.2, this means that the enrollee received exactly one public key response to its requests (and, thus, did not receive the registrar's key), and TEM_RECV_GET never returned OVERLAP or RETRY. As above, there must have been a time when the registrar was listening, and the enrollee transmitted its request message on the registrar's channel. Proposition 5.3 guarantees that the registrar must have either received the enrollee's message, or detected tampering (and returned OVERLAP or RETRY). In both of those cases, Section 5.4.2 requires the registrar to send a reply. Proposition 5.3 similarly guarantees that the enrollee must have either received the registrar's reply or detected tampering (and returned OVERLAP or RETRY), which directly contradicts our supposition. Thus, an adversary cannot convince the enrollee to accept a public key other than the registrar's, and TEP is secure. ■

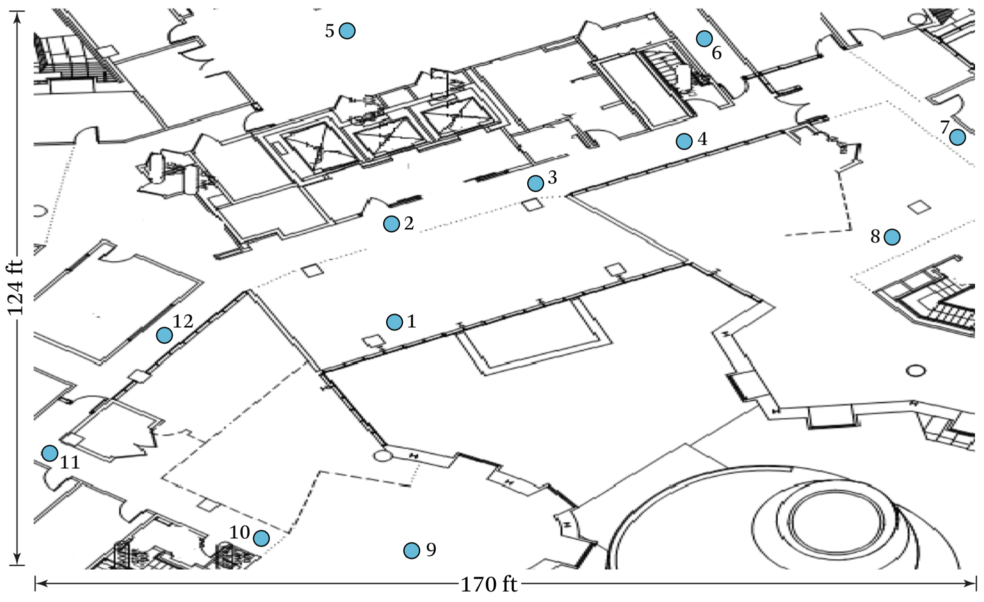


Figure 5.5 **Locations of nodes (indicated by blue circles)** in our experimental testbed, which operates as part of our campus network.

5.6.2 Evaluating TEP's Accuracy

We check whether TEP can be accurately realized using existing operating systems and off-the-shelf 802.11 hardware. Our experiments use our Ath5K prototype described in Section 5.5 and run over our campus network. Figure 5.5 shows the locations of the TEP nodes, which span‘ 21,080 square feet (1958 m^2) with both line-of-sight and non-line-of-sight links.

(a) Transmitter. The performance of TEP hinges on the transmitter accurately scheduling the transmission of the ON–OFF slots. The difficulty in accurate scheduling arises from the fact that we want to implement the protocol in software using standard 802.11 chipsets. Hence, we are limited by the operating system and the hardware interface. For example, if the kernel or the hardware introduces extra delays between the slot packets, it will alter the bit sequence conveyed to the receiver, and will cause failures. Given that our slot is 40 μs, we need an accuracy that is on the order of few microseconds. Can we achieve such an accuracy with existing kernels and chipsets?

Experiment. We focus on the most challenging ON–OFF slot sequence from a scheduling perspective: alternating zeros and ones which requires the maximum scheduling

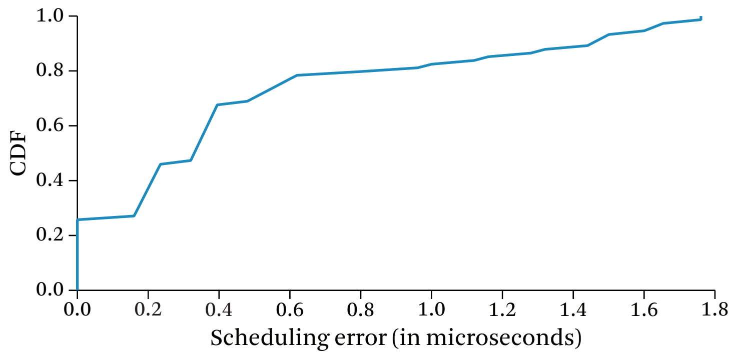


Figure 5.6 **CDF of TEP slot scheduling errors.** The figure shows that the maximum scheduling error is 1.8 μs, which is significantly lower than the slot duration of 40 μs.

precision. We set the slot time to 40 μs, by sending a packet at the highest bitrate of 54 Mbps. To measure the produced slots accurately, we capture the signal transmitted by our 802.11 sender using a USRP2 software radio board. Our USRP2 board can measure signal samples at a resolution of 0.16 μs, allowing us to accurately compute the duration of the produced slots. We run the experiment 1000 times for each sender in our testbed and measure the exact duration of every slot. We then compute the scheduling error as the difference between the measured slot duration and the intended 40 μs.

Results. Figure 5.6 shows the CDF of slot scheduling errors. The figure shows that the median scheduling error is less than 0.4 μs and the maximum error is 1.8 μs. Thus, despite operating in software and with existing chipsets, a TEP sender can accurately schedule the ON–OFF slots at microsecond granularity.

(b) Receiver. TEP's security depends on the receiver's ability to distinguish ON slots from OFF slots. In this section, we check that given that the receiver is within the sender's radio range (i.e., can sense the sender's signal), it can clearly distinguish ON slots from OFF slots.

Experiment. In each run, the sender sends a sequence of alternating ON–OFF slots, using a slot duration of 40 μs. The receiver uses a sensing window of 20 μs to measure fractional occupancy. This means the receiver has twice as many measurements of fractional occupancy as there are slots. As explained in Section 5.5.2, the receiver keeps either the odd or even measurements depending on which sequence has higher variance. Hence, for each slot, the receiver has exactly one fractional occupancy measurement. We then compare the measured fractional occupancy for known ON slots

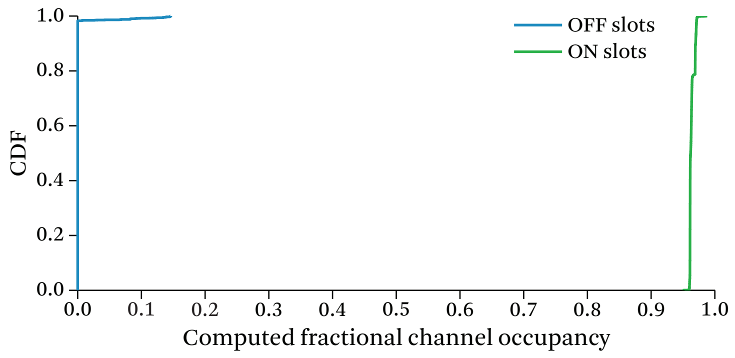


Figure 5.7 **CDFs of the fractional occupancy during ON slots and OFF slots.** The figure shows that the two distributions have no overlap and hence the receiver cannot confuse ON and OFF slots.

vs. known OFF slots to determine if the receiver can reliably distinguish between them based on measured fractional occupancy. We randomly pick two nodes in the testbed to be sender and receiver, and repeat the experiment for various node pairs in the testbed.

Results. Figure 5.7 plots the CDFs of fractional occupancy for ON slots and OFF slots. The figure shows that the two CDFs are completely separate; that is, there is no overlap in the values of fractional occupancy that correspond to OFF slots and those that correspond to ON slots. Hence, by looking at the fractional occupancy the receiver can perfectly distinguish the ON slots from OFF slots. This result shows that a TEP receiver based on current OSes and 802.11 hardware can accurately decode the ON–OFF slots necessary for TEP.

5.6.3 Evaluating TEP's Performance

We are interested in how TEP interacts with cross-traffic in an operational network. Cross-traffic does not hamper TEP's security (the proofs in Section 5.6.1 apply in the presence of cross traffic). However, cross-traffic may cause *false positives*, where a node incorrectly declares that a TEP message has been tampered with by an adversary. Such events can unnecessarily delay secure pairing.

We investigate TEP's interaction with cross-traffic using results from two operational networks: the SIGCOMM 2010 network, which is a heavily congested network, and our campus network, which is a moderately congested network. As in Section 5.6.2, our experiments use our modified Ath5k driver on AR5001X+ Atheros chipsets. In addition to cross-traffic on the TEP channel, both networks carried traffic on adjacent 802.11 channels.

Impact of Cross-Traffic on a Sync Packet. In TEP, a receiver detects a TEM if the medium is continuously occupied for a period longer than the duration of a synchronization packet (19 ms). We would like to check that a receiver is unlikely to encounter false positives while detecting synchronization packets. False positives could occur in two scenarios: either (1) legitimate traffic includes such continuous long bursts of energy, or (2) a TEP receiver is incapable of detecting the short DIFS intervals that separate legitimate packets, and mistakes a sequence of back-to-back WiFi packets as a continuous burst of energy.[7] We empirically study each case below.

Experiment 1. We first check whether legitimate traffic can cause the medium to be continuously occupied for a duration of 19 ms. We use two production networks: our campus network and the SIGCOMM 2010 network. Since we would like to capture all kinds of energy bursts, including collisions, we sense the medium using USRP2 radios. USRP2s allow us to directly look at the signal samples and hence are much more sensitive than 802.11 cards. We used a USRP2 board to eavesdrop on the channel on which these networks operate and log the raw signal samples. In order to compute the length of bursts on the channel, we need to be able to identify the beginning of a burst and its end in an automated way. To do so, we use the double sliding window packet detection algorithm[8] typically used in hardware to detect packet arrivals [Tan 2006]. We collected over a million packets on the SIGCOMM network and about the same number on our campus network. We processed each trace to extract the energy bursts and their durations (as explained above) and plot the CDF of energy burst durations in Figure 5.8.

Result 1. The results in Figure 5.8 show that all energy bursts in both networks lasted for less than 4.3 ms, which is much shorter than a TEP synchronization packet. In particular, the majority of energy bursts last between 0.25 ms and 2 ms. This corresponds to a packet size of 1500 bytes transmitted at a bit rate between 6 Mb/s and 48 Mb/s, which spans the range of 802.11g bit rates. A few bursts lasted for less time which are likely to be short ACK packets. Also, a few bursts have lasted longer than 2 ms. Such

7. A data packet and its ACK are separated by a SIFS, which is smaller than a DIFS, but ACKs are short packets and the next data packet is separated by a DIFS. Hence, the maximum packing occurs with back-to-back data packets without ACKs.

8. The double sliding window algorithm compares the energy in two consecutive sliding windows. If there is no packet, i.e., the two windows are both capturing noise, the ratio of their energy is around 1. Similarly, if both windows are already in the middle of a packet, their relative energy is 1. In contrast, when one window is partially sliding into a packet while the other is still capturing noise, the ratio between their energy starts increasing. The ratio spikes when one window is fully into a packet while the other is still fully in the noise, which indicates that the beginning of the packet is at the boundary between the two windows. Analogously, a steep dip in energy corresponds to the end of a packet [Tan 2006].

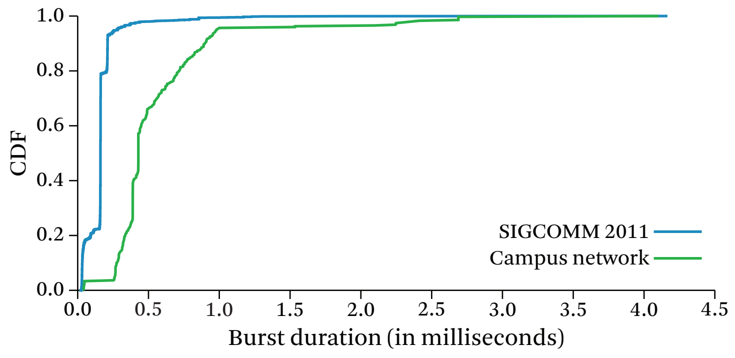


Figure 5.8 **CDF of the duration of energy bursts in the SIGCOMM 2010 network and our campus network.** The figure shows that energy bursts caused by normal traffic are much shorter than a TEP synchronization packet (19 ms). Thus, it is unlikely that TEP will confuse normal traffic as a synchronization packet.

longer bursts are typically due to collisions. Figure 5.9 illustrates this case, where the second packet starts just before the first packet ends, causing a spike in the energy level on the channel. Soon after, the first packet ends, causing the energy to drop again, but the two transmissions have already collided.[9] Interestingly, the bit rates used in our campus network are lower than those used at SIGCOMM. This is likely because at SIGCOMM, the access point was in the conference room and in line-of-sight of senders and receivers, while in our campus, an access point serves multiple offices that span a significant area and are rarely in line-of-sight of the access point.

Overall, the results in Figure 5.8 indicate that bursts of energy in today's production networks have significantly shorter durations than TEP's synchronization packet, and hence are unlikely to cause false positives.

Experiment 2. The second scenario in which a node may incorrectly detect a synchronization packet occurs when the node confuses a sequence of back-to-back packets separated by DIFS as a single continuous energy burst. Thus, we evaluate our prototype's ability to distinguish a synchronization packet from a stream of back-to-back 802.11 packets. To do so, we randomly pick two random nodes in our testbed in Figure 5.5, and make one node transmit a stream of back-to-back 1500-byte packets at the lowest rate of 1 Mbps, while the other node senses the medium using the default sensing window of 2 ms. We then make the same sender transmit a stream of synchro-

9. Collisions of two 1500-byte packets transmitted at 6 Mb/s may be slightly longer than 4 ms because of the additional symbols corresponding to link layer header and trailer, and the PHY layer preamble.

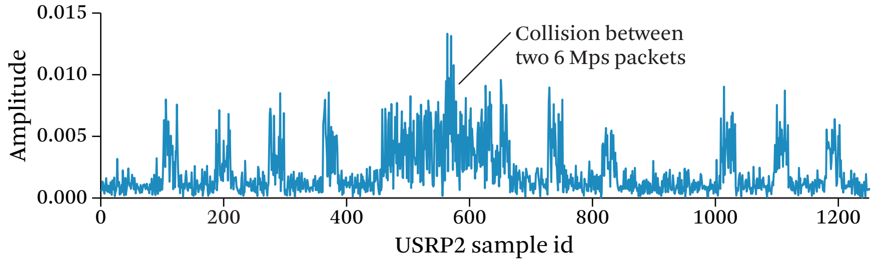


Figure 5.9 **The energy pattern of the maximum energy burst in the SIGCOMM trace.** The figure indicates that such relatively long bursts are due to collisions at the lowest bit rate of 6 Mb/s. The other spikes correspond to packets sent at higher bit rates.

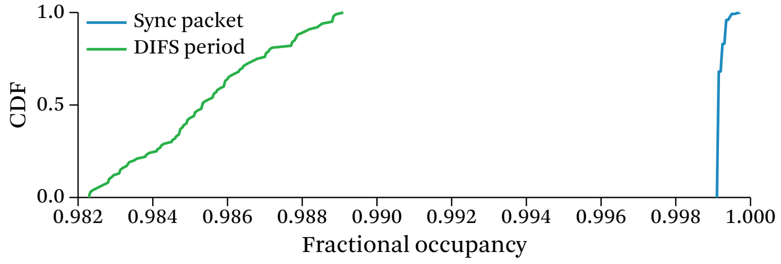


Figure 5.10 **CDF of fractional occupancy measured by a receiver for transmissions of either a synchronization packet or a sequence of back-to-back 1500-byte packets separated by DIFS.** The figure shows a full separation between the two CDFs, indicating that a TEP receiver does not confuse back-to-back packets as a synchronization packet.

nization packets while the receiver senses these packets using a 2 ms window. For both cases, we compute the fractional occupancy in each sensing window. We repeat the experiment with multiple node pairs and compare the occupancy during back-to-back packets and synchronization packets.

Result 2. Figure 5.10 compares the CDF of the fractional occupancy during a synchronization packet and the CDF of the fractional occupancy when the sensing window includes back-to-back packets separated by a DIFS,[10] taken over 100 K synchronization packets and 100 K DIFS occurrences. The figure shows that the two CDFs are

10. Sometimes the DIFS may be split between two consecutive sensing windows, and in this case we include in the CDF whichever of these two window has the lower fractional energy. This is because it is sufficient that one sensing window shows a relatively low fractional occupancy to declare the end of energy burst.

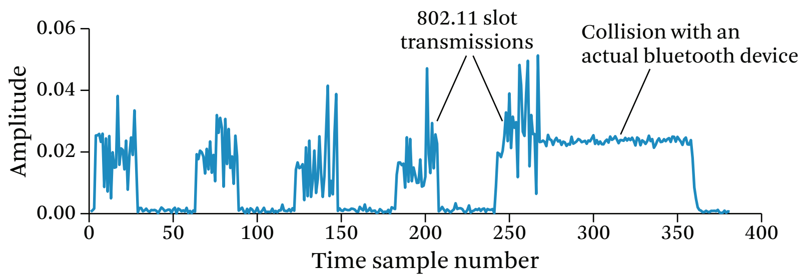


Figure 5.11 **Energy pattern for TEM slots in the presence of a Bluetooth device causing interference.**

sufficiently separate making it unlikely that TEP confuses back-to-back packets as a synchronization packet.

5.6.4 Performance with Non-802.11 Traffic

Finally, while 802.11 nodes comply with the rules of CTS-to-SELF, and abstain from transmitting during TEM's ON–OFF slots, other devices may continue to transmit, causing TEM nodes to detect tampering. Figure 5.11 shows a collision between a TEM and a Bluetooth transmission from an Android phone as captured by a USRP2. Bluetooth devices do not typically decode 802.11 CTS-to-SELF packets, and hence, as shown in the figure, end up transmitting during the ON–OFF slots. In this section we examine the impact of a nearby Bluetooth device on TEM.

Experiment. We place a TEM sender in location 1 (Figure 5.5) and make other nodes act as TEM receivers. We co-locate a Bluetooth device next to the TEM sender. The sender periodically sends an announcement. The receivers first detect the synchronization packets, decode the CTS-to-SELF, and then try to verify the slots. If the receiver can successfully verify, it declares success. Otherwise, it attempts to verify the slots in the next time period.

Results. Figure 5.12 shows the CDF of the number of required attempts before a TEM receiver succeeds in receiving a correct TEM. Bluetooth transceivers operate on 79 bands in 2402–2480 MHz and frequently jump across these bands. Thus, the probability that they interfere with TEM in successive runs of the protocol is relatively low. The figure shows that, even in the presence of Bluetooth devices which cannot decode a CTS-to-SELF, a TEM receiver requires 1.4 attempts on average, and 4 attempts maximum, before it receives the announcement.

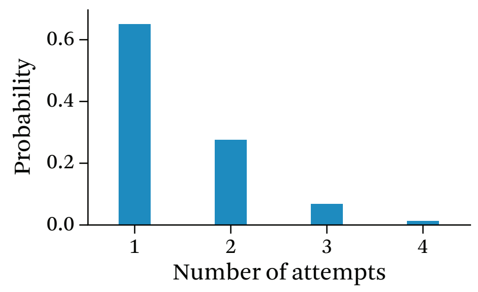


Figure 5.12 **Number of attempts required for TEP to successfully pair in the presence of an interfering Bluetooth device.**

5.7 Discussion

This work presented Tamper-Evident Pairing (TEP), the first wireless pairing protocol that works in-band, with no pre-shared keys, and protects against MITM attacks. TEP relies on a Tamper-Evident message (TEM) mechanism, which guarantees that an adversary cannot tamper with either the payload in a transmitted message, or with the fact that the message was sent. We formally proved that the design protects from MITM attacks. Further, we implemented a prototype of TEM and TEP for the 802.11 wireless protocol using off-the-shelf WiFi devices, and showed that TEP is practical on real-world 802.11 networks and devices.

6 Conclusion

In conclusion, this book is about wireless interference. Traditional systems have regarded interference as an intrinsically harmful phenomenon that must be avoided. Here, we take an alternate approach and show that it is better to understand the nature of interference and incorporate this understanding into the design of protocols and systems. By doing so, we were able to design and build practical systems that transform interference from a harmful to a harmless phenomenon, and even a beneficial phenomenon. Specifically, we make the following contributions:

Decoding 802.11 collisions. In contrast to traditional approaches that try to avoid collisions between wireless devices, this book presents the first 802.11 receiver design that decodes 802.11 collisions, thus rendering them harmless. Our design works without sender modifications and without any assumptions of packet synchronization, large differences in power, or special codes.

Combating high-power cross-technology interference. We design the first WiFi receiver that can decode in the presence of high-power cross-technology interference. We also introduce a new form of cognitive communication where different technologies do not necessarily have to use isolated frequencies, as in traditional cognitive communication, but could in crowded environments use the same frequency band. This enables packing more radios and data in the wireless spectrum.

Non-invasive approach to securing medical implants. We show how to secure insecure medical implants like pacemakers and cardiac defibrillators, without any modification to the implants themselves. To do this, we design the first communication system where, by leveraging interference, the receiver encrypts the transmissions, on behalf of the transmitter. Since our solution does not require modifying existing implants, it helps millions of patients who already have these implants with no cryptography.

Secure pairing without passwords or prior secret keys. We design the first system to establish secure wireless connections without using passwords, prior secret keys, or out-of-band channels. Prior efforts assume that the adversary can arbitrarily create interference and tamper with wireless messages. Thus, they opt for using secret keys or out-of-band channels. In contrast, by understanding interference, we design a wireless message primitive that cannot be altered or hidden without detection. We analytically prove the security of the resulting protocol and empirically demonstrate its practicality.

6.1 Looking Forward

Wireless networking has witnessed a paradigm shift over the last five to seven years. The field has been transformed from treating the physical layer as a black box and having packets as the only interface to the medium, into designing networked systems that tightly incorporate an understanding of the physical layer. This has allowed us to revisit and address classic problems such as hidden terminals and password-free security, and also make a foray into new domains like medical device security.

The next few years are going to be exciting for wireless research because of its ability to change people's lives through diverse applications from smart phones and RFIDs to medical implants and brain-machine interfaces. However, as wireless connectivity gets incorporated into diverse devices and applications, the density of wireless deployments increases. As a result, there is a need to design systems that can address interference at a very large scale (thousands of devices in a small room). While this book takes the first few steps in this direction, addressing this problem at such a large scale in practice remains an open problem. Similarly, on the security side, as critical applications like emergency systems and brain-machine interfaces embrace wireless connectivity, there is an immediate need to design provably secure systems that are resilient to physical layer attacks, including interference and jamming. By embedding a better understanding of interference and the physical layer into our protocols, we can design and build efficient and secure systems that allow wireless networks to fully deliver upon their potential.

Bibliography

Cisco. 20 myths of Wi-Fi interference: Dispel myths to gain high-performing and reliable wireless, White paper C11-44927.1-00, 2007. 49

Ofcom. Estimating the utilisation of key license-exempt spectrum bands, Final Report REP003. Mass Consultants Ltd., April 2009. 6, 49, 52

Farpoint Group. Evaluating interference in wireless LANs: Recommended practice, White paper FPG 2010-135.1. Technical Note, 2010. 1, 6, 49

Miercom. Miercom: Cisco CleanAir competitive testing, Lab Test Report DR100409D. 2010. 49

Cisco. Wireless RF interference customer survey result, White paper C11-609300-00. 2010. 1, 6, 49

J. Åkerberg. State-of-the-art radiosonde telemetry. In *Proc. Symp. Integrated Observing and Assimilation Systems for Atmosphere, Oceans, and Land Surface*. American Meterological Society, 2004. 109

I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: A survey. *Comp. Netw.*, 38(4): 393–422, 2002. DOI: 10.1016/S1389-1286(01)00302-4. 1

J. Andrews. Interference cancellation for cellular systems: A contemporary overview. *Wireless Commun.*, 12(2): 19–29, 2005. DOI: 10.1109/MWC.2005.1421925. 14, 17

E. Aryafar, N. Anand, T. Salonidis, and E. W. Knightly. Design and experimental evaluation of multi-user beamforming in wireless LANs. In *Proc. 14th Annual Int. Conf. on Mobile Computing and Networking*, pages 197–208, 2010. DOI: 10.1145/1859995.1860019. 79

P. Bahl, R. Chandra, T. Moscibroda, R. Murty, and M. Welsh. White space networking with wifi like connectivity. In *Proc. ACM SIGCOMM 2009 Conf. on Data Communication*, pages 27–38, 2009. DOI: 10.1145/1594977.1592573. 80

D. Balfanz, G. Durfee, D.K.Smetters, and R. Grinter. In search of usable security: Five lessons from the field. *IEEE Security and Privacy*, 2(5): 19–24, 2004. DOI: 10.1109/MSP.2004.71. 11, 114, 117, 118

Bandspeed. Understanding the effects of radio frequency (RF) interference on WLAN performance and security. Bandspeed White Paper, 2010. Available from http://www.bandspeed.com/technology/docs/BSP_RF+WLAN_WP.pdf; last retrieved May 2014. 1, 49, 75

V. Bharghavan, A. J. Demers, S. Shenker, and L. Zhang. MACAW: A media access protocol for wireless LAN's. In *Proc. 1994 Conf. on Communication Architectures, Protocols, and Applications*, pages 212–225, 1994. DOI: 10.1145/190809.190334. 3, 4, 5, 17

V. Boyko, P. MacKenzie, and S. Patel. Provably secure password-authenticated key exchange using Diffie-Hellman. In *Proc. Int. Conf. Theory and Application of Cryptographic Techniques*, pages 156–171, 2000. 117

D. G. Brennan. On the maximal signal-to-noise ratio realizable from several noisy signals. *Proc. IRE*, 43:1530, 1955. 20, 30

M. Cagalj, J.-P. Hubaux, S. Capkun, R. Rangaswamy, I. Tsigkogiannia, and M. Srivastava. Integrity codes: Message integrity protection and authentication over insecure channels. In *IEEE Symp. Security and Privacy*, pages 280–294, 2006. DOI: 10.1109/SP.2006.23. 117

L. Cao, L. Yang, and H. Zheng. The impact of frequency-agility on dynamic spectrum sharing. In *IEEE Symp. New Frontiers in Dynamic Spectrum Access Networks*, pages 272–283, 2010. 78, 79

S. Capkun, M. Cagalj, R. Rengaswamy, I. Tsigkogiannis, J.-P. Hubaux, and M. Srivastava. Integrity codes: Message integrity protection and authentication over insecure channels. *IEEE Trans. Dependable and Secure Comp.*, 5(4): 208–223, 2008. DOI: 10.1109/TDSC.2008.11. 10, 11, 114, 117

P. Castoldi. *Multiuser Detection in CDMA Mobile Terminals*. Artech house Publishers, 2002. 17, 22, 34

R. Chandra, R. Mahajan, T. Moscibroda, R. Raghavendra, and P. Bahl. A case for adapting channel width in wireless networks. In *Proc. ACM SIGCOMM 2008 Conf. on Data Communication*, pages 135–146, 2008. DOI: 10.1145/1402946.1402975. 7, 78

Y.-C. Cheng, J. Bellardo, P. Benkö, A. C. S. G. M. Voelker, and S. Savage. Jigsaw: solving the puzzle of enterprise 802.11 analysis. In *Proc. 2006 SIGCOMM Conf. on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 39–50, 2006. DOI: 10.1145/1151659.1159920. 3, 13, 17

S. Cherukuri, K. K. Venkatasubramanian, and S. K. S. Gupta. Biosec: A biometric based approach for securing communication in wireless networks of biosensors implanted in the human body. In *Proc. Int. Conf. Parallel Processing Workshops*, pages 432–439 2003. DOI: 10.1109/ICPPW.2003.1240399. 111

J. Choi, M. Jain, K. Srinivasan, P. Levis, and S. Katti. Achieving single channel, full duplex wireless communication. In *Proc. 16th Annual Int. Conf. on Mobile Computing and*

Networking, pages 1–12, 2010. DOI: 10.1145/1859995.1859997. 9, 48, 83, 90, 100, 102, 111

Intersil Corporation. Isl3873: Wireless LAN integrated medium access controller with baseband processor, 2000. Available from http://pdf1.alldatasheetpt.com/datasheet-pdf/view/109039/INTERSIL/ISL3873.html; last retrieved May 2014. 36, 42

I. Csiszar and J. Korner. Broadcast channels with confidential messages. *IEEE Trans. Information Theory*, 24(3): 339–348, 1978. DOI: 10.1109/TIT.1978.1055892. 111

T. Denning, K. Fu, and T. Kohno. Absence makes the heart grow fonder: New directions for implantable medical device security. In *Proc. 3rd USENIX Workshop on Hot Topics in Security*, Article 5, 2008. 111

W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Trans. Information Theory*, 22(6): 644–654, November 1976. DOI: 10.1109/TIT.1976.1055638. 113

W. Diffie, P. C. van Oorschot, and M. J. Wiener. Authentication and authenticated key exchanges. *Designs, Codes, and Cryptography*, 2(2): 107–125, 1992. DOI: 10.1007/BF00124891. 117

M. Duarte and A. Sabharwal. Full-duplex wireless communications using off-the-shelf radios: Feasibility and first results. In *Conf. Record of 44th Asilomar Conference on Signals, Systems, and Computers*, pages 1558–1562, 2010. DOI: 10.1109/ACSSC.2010.5757799. 9, 48, 83, 111

D. Eckhardt and P. Steenkiste. Measurement and analysis of the error characteristics of an in-building wireless network. In *Proc. SIGCOMM Conf. on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 243–254, 1996. DOI: 10.1145/248157.248178. 105

European Telecommunications Standard Institute. ETSI EN 301 839-1 V1.3.1, 2009. 85

C. Falcon. Inside implantable devices. *Medical Design Tech.*, 2004. 98

Federal Communications Commission. MICS Medical Implant Communication Services, FCC 47CFR95.601-95.673 Subpart E/I Rules for MedRadio Services. Available from http://www.fcc.gov/encyclopedia/medical-device-radiocommunications-service-medradio; last retrieved May 2014. 83, 85, 86, 91, 98

K. Fu. Inside risks: Reducing the risks of implantable medical devices: A prescription to improve security and privacy of pervasive health care. *Comm. ACM*, 52(6): 25–27, 2009. DOI: 10.1145/1516046.1516055. 8, 81

K. Fu. Trustworthy medical device software. In *Public Health Effectiveness of the FDA 510(k) Clearance Process: Measuring Postmarket Performance and Other Select Topics: Workshop Report*. IOM (Institute of Medicine), National Academies Press, 2011. 82, 86

C. L. Fullmer and J. J. Garcia-Luna-Aceves. Solutions to hidden terminal problems in wireless networks. In *Proc. SIGCOMM Conf. on Applications, Technologies, Architectures,* *and Protocols for Computer Communication*, pages 39–49, 1997. DOI: 10.1145/263109.263137. 17

R. G. Gallager. A perspective on multiaccess channels. *IEEE Trans. Information Theory*, 31(2): 124–142, 1985. DOI: 10.1109/TIT.1985.1057022. 2, 14

M. Gast. *802.11 Wireless Networks*. O'Reilly, 2005. 32

A. Goldsmith. *Wireless Communications*. Cambridge University Press, 2005. 90, 94

S. Gollakota, F. Adib, D. Katabi, and S. Seshan. Clearing the RF smog: Making 802.11 robust to cross-technology interference. In *Proc. ACM SIGCOMM 2011 Conf.*, pages 170–181, 2011. DOI: 10.1145/2043164.2018456. 59, 91

S. Gollakota, N. Ahmed, N. Zeldovich, and D. Katabi. Secure in-band wireless pairing. In *Proc. 20th USENIX Security Symp.*, Paper 16, 2011. 124, 125, 127

S. Gollakota, H. Hassanieh, B. Ransford, D. Katabi, and K. Fu. They can hear your heartbeats: Non-invasive security for implanted medical devices. In *Proc. ACM SIGCOMM 2011 Conf.*, pages 2–13, 2011. DOI: 10.1145/2043164.2018438. 87

S. Gollakota and D. Katabi. Zigzag Decoding: Combating Hidden Terminals in Wireless Networks. In *Proc. ACM SIGCOMM 2008 Conf. on Data Communication*, pages 159–170, 2008. DOI: 10.1145/1402946.1402977. 79

S. Gollakota and D. Katabi. Zigzag decoding: Combating hidden terminals in wireless networks. Technical Report MIT-CSAIL-TR-2008-018, MIT, 2008. 30

S. Gollakota and D. Katabi. Physical layer wireless security made fast and channel-independent. In *Proc. 30th IEEE Int. Conf. on Computer Communications*, pages 1125–1133, 2011. DOI: 10.1109/INFCOM.2011.5934889. 111

S. Gollakota, S. D. Perli, and D. Katabi. Interference alignment and cancellation. In *Proc. ACM SIGCOMM 2009 Conf. on Data Communication*, pages 159–170, 2009. DOI: 10.1145/1594977.1592588. 68, 79

M. T. Goodrich, M. Sirivianos, J. Solis, G. Tsudik, and E. Uzun. Loud and clear: human-verifiable authentication based on audio. In *Proc. 26th Int. Conf. on Distributed Computing Systems*, page 10, 2006. DOI: 10.1109/ICDCS.2006.52. 10, 11, 114, 117, 118

R. Gummadi, D. Wetherall, B. Greenstein, and S. Seshan. Understanding and mitigating the impact of RF interference on 802.11 networks. In *Proc. 2007 SIGCOMM Conf. on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 385–396, 2007. DOI: 10.1145/1282427.1282424. 17

J. D. Halamka. Telemonitoring for the home, *Life as a Healthcare CIO*, 10 April 2010. Available from http://geekdoctor.blogspot.ca/2010/04/telemonitoring-for-home.html; last retrieved May 2014. 114

D. Halperin, J. Ammer, T. Anderson, and D. Wetherall. Interference cancellation: Better receivers for a new wireless MAC. In *Proc. 7th ACM Workshop on Hot Topics in Networks*, 2007. 16, 79

D. Halperin, T. Anderson, and D. Wetherall. Taking the sting out of carrier sense: interference cancellation for wireless LANs. In *Proc. 14th Annual Int. Conf. on Mobile Computing and Networking*, pages 339–350, 2008a. DOI: 10.1145/1409944.1409983. 5, 16, 17

D. Halperin, T. S. Heydt-Benjamin, K. Fu, T. Kohno, and W. H. Maisel. Security and privacy for implantable medical devices. *IEEE Pervasive Computing*, 7(1): 30–39, 2008b. 8, 9, 81, 110

D. Halperin, T. S. Heydt-Benjamin, B. Ransford, S. S. Clark, B. Defend, W. Morgan, K. Fu, T. Kohno, and W. H. Maisel. Pacemakers and implantable cardiac defibrillators: Software radio attacks and zero-power defenses. In *Proc. IEEE Symp. on Security and Privacy*, pages 129–142, 2008c. 81, 82, 85, 99, 108, 110

Y. He, J. Fang, J. Zhang, H. Shen, K. Tan, and Y. Zhang. MPAP: Virtualization architecture for heterogenous wireless APs. In *Proc. ACM SIGCOMM 2010 Conf.*, pages 133–134, 2010. DOI: 10.1145/1851275.1851271. 78

J. Heiskala and J. Terry. *OFDM Wireless LANs: A Theoretical and Practical Guide*. Sams Publishing, 2001. 57, 65

J. Hou, J. Smee, H. D. Pfister, and S. Tomasin. Implementing interference cancellation to increase the ev-do rev a reverse link capacity. *IEEE Commun. Mag.*, 44(2): 58–64, 2006. DOI: 10.1109/MCOM.2006.1593551. 5, 14, 17

IEEE. 802.15.1 specification: Personal area networks, 2002. 117

IEEE. 802.11i specification: Amendent 6: MAC security enhancements, 2004. 117

Industry Canada. Radio Standards Specification RSS-243: Medical Devices Operating in the 401–406 MHz Frequency Band. Spectrum Management and Telecommunications, 2010. 85

International Telecommunications Union. ITU-R Recommendation RS.1346: Sharing between the meteorological aids service and medical implant communication systems (MICS) operating in the mobile service in the frequency band 401–406 MHz, 1998. 85, 91

K. Jamieson and H. Balakrishnan. PPR: Partial Packet Recovery for Wireless Networks. In *Proc. 2007 SIGCOMM Conf. on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 409–420, 2007. DOI: 10.1145/1282427.1282426. 65, 79

G. Judd and P. Steenkiste. Using emulation to understand and improve wireless networks and applications. In *Proc. 2nd USENIX Symp. on Networked Systems Design & Implementation*, pages 203–216, 2005. 3, 4, 13, 17, 18, 40

P. Karn. Maca–a new channel access method for packet radio. *Proc. 9th ARRL Computer Networking Conf.*, pages 134–140, 1990 3, 4, 5, 17

S. Katti, S. Gollakota, and D. Katabi. Embracing wireless interference: Analog network coding. In *Proc. 2007 SIGCOMM Conf. on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 397–408, 2007. DOI: 10.1145/1282427.1282425. 17, 79

Kelton Research. *Survey: Protecting wireless network an essential element of home security*, November 2006. 10, 113, 118

S. Khurana, A. Kahol, and A. P. Jayasumana. Effect of hidden terminals on the performance of ieee 802.11 MAC protocol. *Proc. 23rd Annual Conf. on Local Computer Networks*, pages 12–20, 1998. DOI: 10.1109/LCN.1998.727642. 3, 13, 17, 18

M. Koplow, A. Chen, D. Steingart, P. Wright, and J. Evans. Thick film thermoelectric energy harvesting systems for biomedical applications. In *Proc. 5th Int. Summer School and Symp. on Medical Devices and Biosensors*, pages 322–325, 2008. DOI: 10.1109/ISSMDBS.2008.4575084. 81, 110

C. Kuo, J. Walker, and A. Perrig. Low-cost manufacturing, usability and security: An analysis of bluetooth simple pairing and wi-fi protected setup. In *Proc. 11th Int. Conf. on Financial Cryptography and 1st Int. Conf. on Usable Security*, pages 325–340, 2007. DOI: 10.1007/978-3-540-77366-5_30. 114

K. Lakshminarayanan, S. Sapra, S. Seshan, and P. Steenkiste. RFDump: An architecture for monitoring the wireless ether. In *Proc. 5th Int. Conf. on Emerging Networking Experiments and Technologies*, pages 253–264, 2009. DOI: 10.1145/1658939.1658968. 49, 79

E. A. Lee and D. G. Messerschmitt. *Digital Communications*. Boston: Kluwer Academic, 1988. 22

J. Lee, W. Kim, S.-J. Lee, D. Jo, J. Ryu, T. Kwon, and Y. Choi. An experimental study on the capture effect in 802.11a networks. *Proc. 2nd ACM Int. Workshop on Wireless Network Testbeds, Experimental Evaluation and Characterization*, pages 19–26, 2007. DOI: 10.1145/1287767.1287772. 40

L. E. Li, K. Tan, Y. Xu, H. Visvanathan, and R. Yang. Retransmission $\neq$ repeat: simple retransmission permutation can resolve overlapping channel collisions. In *Proc. 16th Annual Int. Conf. on Mobile Computing and Networking*, pages 281–292, 2010. DOI: 10.1145/1859995.1860028. 48

T. Li, M. K. Han, A. Bhartia, L. Qiu, E. Rozner, Y. Zhang, and B. Zarikoff. CRMA: Collision-resistant multiple access. In *Proc. 17th Annual Int. Conf. on Mobile Computing and Networking*, pages 61–72, 2011. DOI: 10.1145/2030613.2030621. 48

K. Lin, S. Gollakota, and D. Katabi. Random access heterogenous MIMO networks. In *Proc. ACM SIGCOMM 2011 Conf.*, pages 146–157, 2011. DOI: 10.1145/2043164.2018454. 48

Linksys. Broadcom wireless LAN adapter user guide. Available from http://www.linksysinfo.org/index.php?threads/broadcom-wireless-lan-adapter-user-guide.15819/; last retrieved May 2014. 13

Y. Liu, P. Ning, H. Dai, and A. Liu. Randomized differential DSSS: Jamming-resistant wireless broadcast communication. In *Proc. 29th IEEE Int. Conf. on Computer Communications*, pages 1–9, 2010. DOI: 10.1109/INFCOM.2010.5462156. 111

J. Lopatka. Adaptive generating of the jamming signal. In *Proc. IEEE Military Communications Conf.*, Vol. 2, pages 557–559, 1995. DOI: 10.1109/MILCOM.1995.483528. 93

W. H. Maisel. Safety issues involving medical devices: Implications of recent implantable cardioverter-defibrillator malfunctions. *J. American Medical Association*, 294(8): 955–958, 2005. 8, 81, 82

W. H. Maisel and T. Kohno. Improving the security and privacy of implantable medical devices. *N Engl J Med.*, 362(13): 1164–1166, 2010. 86

I. Martinovic, P. Pichota, and J. Schmitt. Jamming for good: A fresh approach to authentic communication in WSNs. In *Proc. ACM Conf. on Wireless Network Security*, pages 161–168, 2009. DOI: 10.1145/1514274.1514298. 111

R. Mayrhofer and H. Gellersen. Shake well before use: Authentication based on accelerometer data. In *Proc. 5th Int. Conf. on Pervasive Computing*, pages 144–161, 2007. 10, 11, 114, 117, 118

J. M. McCune, A. Perrig, and M. K. Reiter. Seeing-is-believing: using camera phones for human-verifiable authentication. In *Proc. IEEE Symp. Security and Privacy*, pages 110–124, 2005. DOI: 10.1109/SP.2005.19. 10, 11, 114, 117, 118

G. Ostrovsky. Medtronic's Paradigm Veo wireless insulin pump helps prevent hypoglycemia. *MedGadget—Internet Journal for emerging medical technologies*, 2009. Available from http://www.medgadget.com/2009/09/medtronics_paradigm_veo_wireless_insulin_pump_cgm_system_helps_prevent_hypoglycemia.html; last retrieved May 2014. 86

H. Meyr, M. Moeneclaey, and S. A. Fechtel. *Digital Communication Receivers: Synchronization, Channel Estimation, and Signal Processing*. John Wiley & Sons, 1998. 20, 21, 26, 27, 34, 102, 104

S. Mishra, R. Brodersen, S. Brink, and R. Mahadevappa. Detect and avoid: An ultra-wideband/WiMAX coexistence mechanism. *IEEE Commun. Mag.*, 45(6): 68–75, 2007. DOI: 10.1109/MCOM.2007.374435. 78

T. Moscibroda, R. Chandra, Y. Wu, S. Sengupta, P. Bahl, and Y. Yuan. Load-aware spectrum distribution in wireless LANs. In *Proc. 2008 IEEE Int. Conf. on Network Protocols*, pages 137–146, 2008. DOI: 10.1109/ICNP.2008.4697032. 49, 79

A. Muqattash and M. Krunz. CDMA-based MAC protocol for wireless ad hoc networks. In *Proc. 4th ACM Int. Symp. Mobile ad hoc Networking & Computing*, pages 153–164, 2003. DOI: 10.1145/778415.778434. 17

Netgear. Reference manual for the NETGEAR prosafe 802.11g wireless ap WG102. Available from http://documentation.netgear.com/wg102/enu/202-10144-01/pdfs/FullManual.pdf; last retrieved May 2014. 4, 13

P. C. Ng, S. C. Liew, K. C. Sha, and W. T. To. Experimental study of hidden node problem in IEEE 802.11 wireless networks. In *Proc. 2005 SIGCOMM Conf. on Applications, Technologies, Architectures, and Protocols for Computer Communication Poster*, 2005. Available from http://conferences.sigcomm.org/sigcomm/2005/poster-110.pdf; last retrieved May 2014. 3, 13, 17

D. A. Norman. The way I see it: When security gets in the way. *Interactions*, 16(6): 60–63, 2009. DOI: 10.1145/1620693.1620708. 10, 113, 118

D. Panescu. Wireless communication systems for implantable medical devices. *IEEE Eng. in Medicine and Biology Mag.*, 27(2): 96–101, 2008. DOI: 10.1109/EMB.2008.915488. 8, 81, 95

PCTEST Engineering Labs, Inc. Certificate of compliance, FCC part 95 certification, test report number 95.220719375.lf5, 2002. 100, 102

PCTEST Engineering Labs, Inc. Certificate of compliance, FCC part 95 and en 301 839-2, test report number 0703090168.med, 2007. 100, 102

C. Pöpper, M. Strasser, and S. Čapkun. Jamming-resistant broadcast communication without shared keys. In *USENIX Security Sym.*, 2009. 111

B. Radunovic, D. Gunawardena, P. Key, A. Proutiere, N. Singh, H. V. Balan, and G. Dejean. Rethinking indoor wireless: Low power, low frequency, full-duplex. Technical report, Microsoft Research, 2009. 90

H. Rahul, N. Kushman, D. Katabi, C. Sodini, and F. Edalat. Learning to Share: Narrowband-Friendly Wideband Networks. In *Proc. ACM SIGCOMM 2008 Conf. on Data Communication*, pages 147–158, 2008. DOI: 10.1145/1402946.1402976. 7, 49, 78

C. Reis, R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan. Measurement-based models of delivery and interference. In *Proc. 2006 SIGCOMM Conf. on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 51–62, 2006. DOI: 10.1145/1151659.1159921. 17

M. Rieback, B. Crispo, and A. Tanenbaum. RFID Guardian: A battery-powered mobile device for RFID privacy management. In *Proc. 10th Australasian Conf. on Information Security and Privacy*, pages 184–194, 2005. 111

V. Roth, W. Polak, E. Rieffel, and T. Turner. Simple and effective defense againgst evil twin access points. In *Proc. 1st ACM Conf. Wireless Network Security*, pages 220–235, 2008. DOI: 10.1007/11506157_16. 10, 11, 114, 117

Rysavy Research. Mobile broadband capacity constraints and the need for optimization. Report, 2010. Available from http://www.rysavy.com/Articles/2010_02_Rysavy_Mobile_Broadband_Capacity_Constraints.pdf; last retrieved May 2014. 1

D. Sagan. Rf integrated circuits for medical applications: Meeting the challenge of ultra low power communication. Report, 2007. 85

N. Santhapuri, R. R. Choudhury, J. Manweiler, S. Nelakuduti, S. Sen, and K. Munagala. Message in message (MIM): A case for reordering transmissions in wireless networks. In *Proc. 7th ACM Workshop on Hot Topics in Networks*, 2008. 87

K. Sayrafian-Pour, W. Yang, J. Hagedorn, J. Terrill, K. Yazdandoost, and K. Hamaguchi. Channel models for medical implant communication. *Int. J. Wireless Inf. Networks*, 17:105–112, 2010. DOI: 10.1109/PIMRC.2009.5449869. 97

S. Schechter. Security that is meant to be skin deep: Using ultraviolet micropigmentation to store emergency-access keys for implantable medical devices. In *Proc. 1st USENIX Workshop on Health Security and Privacy*, 2010. 9, 111

M. Scheffler, E. Hirt, and A. Caduff. Wrist-wearable medical devices: Technologies and applications. *Medical Device Technology*, 14(7): 26–30, 2003. 86

S. Sen, R. R. Choudhury, and S. Nelakuditi. CSMA/CN: Carrier sense multiple access with collision notification. In *Proc. 16th Annual Int. Conf. on Mobile Computing and Networking*, pages 25–36, 2010. DOI: 10.1109/TNET.2011.2174461. 48

C. E. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, 28(4): 656–715, 1949. 91

W.-L. Shen, Y.-C. Tung, K.-C. Lee, K. Lin, S. Gollakota, D. Katabi, and M.-S. Chen. Rate adaptation for 802.11 multiuser MIMO networks. In *Proc. 18th Annual Int. Conf. on Mobile Computing and Networking*, Pages 29–40, 2012. DOI: 10.1145/2348543.2348551. 48

V. Shnayder, B. Chen, K. Lorincz, T. R. F. Fulford-Jones, and M. Welsh. Sensor networks for medical care. Technical Report TR-08-05, Harvard University, 2005. DOI: 10.1145/1098918.1098979. 81, 110

M. J. Siavoshani, U. Pulleti, E. Atsan, I. Safaka, C. Fragoulia, K. Argyraki, and S. Diggavi. Exchanging secrets without using cryptography. *arXiv:1105.4991v1*, 2011. Available from http://arxiv.org/abs/1105.4991v1; last retrieved May 2014. 111

F. Stajano and R. Anderson. The Resurrecting Duckling: Security issues for ad-hoc wireless networks. In *Proc. 7th Int. Workshop on Security Protocols*, pages 172–182, 1999. 10, 11, 114, 117, 118

T. Taher, M. Misurac, J. LoCicero, and D. Ucci. Microwave oven signal modelling. In *Proc. IEEE Wireless Communications & Networking Conf.*, pages 1235–1238, 2008. DOI: 10.1109/WCNC.2008.222. 67, 75, 79

J. K. Tan. An adaptive orthogonal frequency division multiplexing baseband modem for wideband wireless channels. Master's thesis, MIT, 2006. Available from http://dspace.mit.edu/bitstream/handle/1721.1/37942/144571690.pdf; last retrieved May 2014. 36, 42, 139

K. Tan, H. Liu, J. Fang, W. Wang, J. Zhang, M. Chen, and G. M. Voelker. SAM: Enabling practical spatial multiple access in wireless LAN. In *Proc. 15th Annual Int. Conf. on Mobile Computing and Networking*, pages 49–60, 2009. DOI: 10.1145/1614320.1614327. 62, 79

D. Tse and P. Vishwanath. *Fundamentals of Wireless Communications*. Cambridge University Press, 2005. 1, 2, 14, 17, 42, 51, 67, 79, 86, 90, 95

D. Tse, P. Viswanath, and L. Zheng. Diversity-multiplexing tradeoff in multiple access channels. *IEEE Trans. Information Theory*, 50(9): 1859-1874, 2004. DOI: 10.1109/TIT.2004.833347. 17

S. Verdu. *Multiuser Detection*. Cambridge University Press, 1998. 1, 14, 17, 20

A. J. Viterbi. Very low rate convolutional codes for maximum theoretical performance of spread-spectrum multiple-access channels. *IEEE J. Selected Areas in Comm.*, 8(4): 641–649, 1990. DOI: 10.1109/49.54460. 17

M. Vutukuru, H. Balakrishnan, and K. Jamieson. Cross-layer wireless bit rate adaptation. In *Proc. ACM SIGCOMM 2009 Conf. on Data Communication*, pages 3–14, 2009. DOI: 10.1145/1594977.1592571. 65

J. Wang, H. Hassanieh, D. Katabi, and P. Indyk. Efficient and reliable low-power backscatter networks. In *Proc. ACM SIGCOMM 2012 Conf.*, pages 61–72, 2012. 48

C. Ware, J. Judge, J. Chicharo, and E. Dutkiewicz. Unfairness and capture behavior in 802.11 adhoc networks. In *IEEE Int. Conf. on Communications*, Volume 1, pages 159–163, 2000. DOI: 10.1109/ICC.2000.853084. 3, 4, 13, 17, 18, 114

IEEE. 802.11-2012—IEEE Standard for Information technology—Telecommunications and information exchange between systems Local and metropolitan area networks–Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. 2012. Available from http://standards.ieee.org/about/get/802/802.11.html; last retrieved May 2014. 14, 22, 24, 31, 65, 69

WiFi Alliance. WiFi protected setup specification, version 1.0h, 2006. 10, 113, 114, 118, 120

G. Woo, P. Kheradpour, and D. Katabi. Beyond the bits: Cooperative packet recovery using physical layer information. In *Proc. 13th Annual Int. Conf. on Mobile Computing and Networking*, pages 147–158, 2007. DOI: 10.1145/1287853.1287871. 30, 32

A. Wyner. The wire-tap channel. *Bell Sys. Technical Journal*, 54(8): 1355–1387, 1975. 111

F. Xu, Z. Qin, C. C. Tan, B. Wang, and Q. Li. IMDGuard: Securing implantable medical devices with the external wearable guardian. In *Proc. 30th IEEE Int. Conf. on Computer Communications*, pages 1862–1870, 2011. DOI: 10.1109/INFCOM.2011.5934987. 111

K. Xu, M. Gerla, and S. Bae. Effectiveness of RTS/CTS handshake in IEEE 802.11 based ad hoc networks. *In Ad Hoc Network Journal*, 1(1): 107–123, 2003. DOI: 10.1109/GLOCOM.2002.1188044. 13

L. Yang, W. Hou, L. Cao, B. Y. Zhao, and H. Zheng. Supporting demanding wireless applications with frequency-agile radios. In *Proc. 7th USENIX Symp. on Networked Systems Design & Implementation*, Article 5, 2010. 49, 78

P. Zetterberg. Experimental investigation of TDD reciprocity-based zero-forcing transmit precoding. *EURASIP J. Adv. Signal Process*, 2011: Article ID 137541, 2011. DOI: 10.1155/2011/137541. 68

C. Zhan, W. B. Baine, A. Sedrakyan, and S. Claudia. Cardiac device implantation in the US from 1997 through 2004: A population-based analysis. *Journal of General Internal Medicine*, 23:13–19, 2007. 8, 81

H. Zhao, Y. Q. Shi, and N. Ansari. Hiding data in multimedia streaming over networks. In *Proc. 8th Annual Communication Networks and Services Research Conf.*, pages 50–55, 2010. DOI: 10.1109/CNSR.2010.20.

J. Zhu, X. Guo, S. Roy, and K. Papagiannaki. CSMA self-adaptation based on interference differentiation. In *Proc. IEEE Global Telecommunications Conf.*, pages 4946–4951, 2007. DOI: 10.1109/GLOCOM.2007.938. 17

Author's Biography

Shyamnath Gollakota

Shyamnath Gollakota is an assistant professor in Computer Science and Engineering at the University of Washington. His research focuses on designing and building novel wireless systems. His work has been awarded the ACM Doctoral Dissertation Award 2012, ACM SIGCOMM 2008 Best paper award for ZigZag decoding, ACM SIGCOMM 2011 Best Paper Award for securing medical implants, ACM SIGCOMM 2013 Best Paper Award for ambient backscatter, ACM MOBICOM 2013 Best Paper award for Wi-Fi gesture recognition, and AT&T Applied Security Award for password-free wireless security. His work has appeared in venues like the BBC, *Forbes,* NBC News, Network World, the *New York Times,* Slashdot, the *Washington Post,* and *Wired*. He received his Ph.D. in Electrical Engineering and Computer Science at MIT, and a bachelors degree in Computer Science and Engineering at IIT Madras.