2008

# Pacemakers and Implantable Cardiac Defibrillators: Software Radio Attacks and Zero-Power Defenses

Daniel Halperin
*University of Washington*

# Pacemakers and Implantable Cardiac Defibrillators: Software Radio Attacks and Zero-Power Defenses

Daniel Halperin†
University of Washington

Thomas S. Heydt-Benjamin†
University of Massachusetts Amherst

Benjamin Ransford†
University of Massachusetts Amherst

Shane S. Clark
University of Massachusetts Amherst

Benessa Defend
University of Massachusetts Amherst

Will Morgan
University of Massachusetts Amherst

Kevin Fu, PhD*
University of Massachusetts Amherst

Tadayoshi Kohno, PhD*
University of Washington

William H. Maisel, MD, MPH*
BIDMC and Harvard Medical School

***Abstract*—Our study analyzes the security and privacy properties of an implantable cardioverter defibrillator (ICD). Introduced to the U.S. market in 2003, this model of ICD includes pacemaker technology and is designed to communicate wirelessly with a nearby external programmer in the 175 kHz frequency range. After partially reverse-engineering the ICD's communications protocol with an oscilloscope and a software radio, we implemented several software radio-based attacks that could compromise patient safety and patient privacy. Motivated by our desire to improve patient safety, and mindful of conventional trade-offs between security and power consumption for resource-constrained devices, we introduce three new zero-power defenses based on RF power harvesting. Two of these defenses are human-centric, bringing patients into the loop with respect to the security and privacy of their implantable medical devices (IMDs). Our contributions provide a scientific baseline for understanding the potential security and privacy risks of current and future IMDs, and introduce human-perceptible and zero-power mitigation techniques that address those risks. To the best of our knowledge, this paper is the first in our community to use general-purpose software radios to analyze and attack previously unknown radio communications protocols.**

## I. Introduction

Wirelessly reprogrammable *implantable medical devices* (*IMDs*) such as *pacemakers*, *implantable cardioverter defibrillators* (*ICDs*), *neurostimulators*, and implantable *drug pumps* use embedded computers and radios to monitor chronic disorders and treat patients with automatic therapies. For instance, an ICD that senses a rapid heartbeat can administer an electrical shock to restore a normal heart rhythm, then later report this event to a health care practitioner who uses a *commercial device programmer*[1] with wireless capabilities to extract data from the ICD or modify its settings without surgery. Between 1990 and 2002, over 2.6 million pacemakers and ICDs were implanted in patients in the United States [19]; clinical trials have shown that these devices significantly improve survival rates in certain populations [18]. Other research has discussed potential security and privacy risks of IMDs [1], [10], but we are unaware of any rigorous public investigation into the observable characteristics of a real commercial device. Without such a study, it is impossible for the research community to assess or address the security and privacy properties of past, current, and future devices. We address that gap in this paper and, based on our findings, propose and implement several prototype attack-mitigation techniques.

Our investigation was motivated by an interdisciplinary study of medical device safety and security, and relied on a diverse team of area specialists. Team members from the security and privacy community have formal training in computer science, computer engineering, and electrical engineering. One team member from the medical community is a practicing cardiologist with hundreds of pacemaker and implantable defibrillator patients and was past chairperson of the FDA's Circulatory System Medical Device Advisory Panel. Our technical contributions toward understanding and improving the security, privacy, and safety of these devices include: analyses; software radio-based methodologies; and human-perceptible and zero-power (battery-free) defenses.

**Overview of contributions.** We assess the security and privacy properties of a common ICD and present attacks on privacy, integrity, and availability. We show that the ICD discloses sensitive information in the clear (unencrypted); we demonstrate a *reprogramming* attack that changes the operation of (and the information contained in) the ICD; and

*Corresponding faculty authors:

- Kevin Fu, Medical Device Security Center, Department of Computer Science, University of Massachusetts Amherst, 140 Governors Drive, Amherst, Massachusetts 01003 (kevinfu@cs.umass.edu);
- Tadayoshi Kohno, Medical Device Security Center, Department of Computer Science and Engineering, University of Washington, Box 352350, Seattle, Washington 98195 (yoshi@cs.washington.edu);
- William H. Maisel, Medical Device Safety Institute, Beth Israel Deaconess Medical Center, Harvard Medical School, 185 Pilgrim Road, Baker 4, Boston, MA 02215 (wmaisel@bidmc.harvard.edu).

Additional information online at http://www.secure-medicine.org.

†Co-student leads listed in alphabetical order; each participated equally.

[1]The reader should not confuse the term "device programmer" with a person who programs computers. The former is an external device that communicates with and adjusts the settings on an IMD.

we give evidence that a battery-powered ICD can be made to communicate indefinitely with an unauthenticated device, thereby posing a potential denial-of-service risk. All of our attacks can be mounted by an unauthorized party equipped with a specially configured radio communicator within range of the ICD. In our experiments, we performed attacks from distances up to several centimeters; we did not experiment with increasing this range. We also present prototype defenses against the attacks we describe.

Developing these attacks and defenses required some understanding of relevant devices and protocols. We give an account of our attempts to reverse-engineer communications to and from the ICD using a commercial ICD programmer and a *software radio*; we believe that this work is the first in our community to successfully use general purpose software radios in the reverse engineering of wireless protocols for security analysis. Information directly available from RF signals, given knowledge gained from reverse engineering, includes patient information (such as name and diagnosis) and medical telemetry (information about vital signs); we demonstrate this breach of privacy by showing redacted transmissions. We also demonstrate active attacks on device integrity wherein an unauthorized software radio transmitter (or unauthorized external programmer) that follows a certain protocol can change information on the ICD, including therapy settings that determine when the device should administer an electric shock; the unauthorized radio transmitter can also make the ICD issue a commanded electrical shock. Finally, we describe our discovery that an attacker can keep an ICD in a state of elevated energy consumption, thereby potentially depleting battery life and threatening availability. Table I provides a summary of our attacks and their implications.

With the above attacks in mind, we present prototype defenses against them. Our defenses comprise three different deterrence and prevention mechanisms that sit at the interface between an ICD and the outside world. Our defenses do not require battery power and therefore may require only minimal design changes to future implantable devices. We refer to these mechanisms as *zero-power* defenses to emphasize that they draw no energy from the primary battery, instead harvesting RF energy from an external source. *Zero-power notification* audibly warns a patient of security-sensitive events and can help mitigate the risk of attacks both by outsiders who have custom equipment (like our software radio-based attack system) and by insiders who have access to commercial ICD programmers. *Zero-power authentication* uses symmetric cryptographic techniques to prevent unauthorized access; it aims to protect against adversaries who have custom equipment. Finally, *sensible security* combines elements of zero-power notification and authentication by allowing patients to physically sense an acoustic key exchange. We show the effectiveness of these defenses by measuring the performance of our prototype zero-power system, *WISPer*, which is a WISP UHF RFID tag [25], [27] augmented with a piezo-element. (Notationally, although WISPer is batteryless and is based on RFID technologies, the term *passive* does not accurately characterize the radio conversation that takes place.)

Our study examines a single *Medtronic Maximo DR VVE-DDDR* model #7278 ICD. This model, introduced to the U.S. market in 2003 [21], is a typical ICD: it incorporates pacemaking (steady, periodic electrical stimulation) and defibrillation (single large shock) functions, and it communicates with an external device programmer at a range of several centimeters.

**Implications, challenges, and broader issues.** Our study focuses on a single ICD, and therefore provides only a small snapshot in the evolution and breadth of ICD technologies and more general implantable medical devices. Nevertheless, we believe that this snapshot is necessary toward assessing the current trajectory of IMD security and privacy. We hope that the analyses and defenses presented in this paper will motivate broader scientific investigations into how to best provide security, privacy, safety, and effectiveness for future implantable medical devices. Improving IMD security and privacy is, however, significantly challenging due to rapidly evolving threat models, trends toward longer-range wireless communication, explorations into multi-agent systems of intercommunicating IMDs [4], [7], [33], and resource constraints of an IMD's battery, processor, and memory. Moreover, as we previously observed [10], there is tension between security (restricted access) and safety (open access in emergency scenarios); the *zero-power notification* portion of our WISPer prototype aims to address this tension.

**Attack scenarios.** Since health care is a very sensitive and personal subject for many people, we explicitly choose to deviate from standard practice in the academic security research community and *do not* describe specific scenarios in which an attacker might compromise the privacy or health of a victim. We also do not discuss the potential impact on patients if an adversary were to carry out an attack *in vivo*. Rather, when discussing attacks we focus solely on the technical properties of those attacks. In addition, in each case where we identify a vulnerability, we propose a solution or technical direction to mitigate it.

**Context.** Pacemakers, ICDs, and other implantable medical devices have improved and saved innumerable lives. To our knowledge, no IMD patient has ever been harmed by a malicious security attack. While our research demonstrates that such a scenario is possible, our goals in conducting this research are to: (1) demonstrate that IMD security and privacy vulnerabilities exist; (2) propose solutions to the identified weaknesses; (3) encourage the development of more robust security and privacy features for IMDs; and (4) improve the privacy and safety of IMDs for the millions of patients who enjoy their benefits. This paper, which focuses on a single ICD and our zero-power defenses, should be read in concert with our previous work [10], which surveys the potential security and privacy issues for broad classes of IMDs independent of any particular IMD technology.

**Disclosure.** In light of the rapid advances and changes in ICD technology, we conducted this scientific investigation with the

| | Commercial programmer | Software radio eavesdropper | Software radio programmer | Primary risk |
|---|---|---|---|---|
| Determine whether patient has an ICD | ✔ | ✔ | ✔ | Privacy |
| Determine what kind of ICD patient has | ✔ | ✔ | ✔ | Privacy |
| Determine ID (serial #) of ICD | ✔ | ✔ | ✔ | Privacy |
| Obtain private telemetry data from ICD | ✔ | ✔ | ✔ | Privacy |
| Obtain private information about patient history | ✔ | ✔ | ✔ | Privacy |
| Determine identity (name, etc.) of patient | ✔ | ✔ | ✔ | Privacy |
| Change device settings | ✔ | | ✔ | Integrity |
| Change or disable therapies | ✔ | | ✔ | Integrity |
| Deliver command shock | ✔ | | ✔ | Integrity |

TABLE I
RESULTS OF EXPERIMENTAL ATTACKS. A CHECK MARK INDICATES A SUCCESSFUL IN VITRO ATTACK.

goal of understanding and addressing the potential security risks of ICDs before future ICDs and other IMDs become more complex and the potential security and privacy risks to patients increase. However, we also firmly believe in disclosing this information in an ethical manner that fully considers the well-being of patients. We specifically and purposefully omit details that would allow someone to use this article as a guide for creating attacks against ICDs.

**Paper organization.** Section II gives a brief introduction to ICDs, describes the security model we consider in this work, and summarizes related work. Section III discusses the process of intercepting and reverse-engineering an ICD's wireless communications, beginning with RF signal analysis and culminating in readable plaintext. Section IV discusses replay attacks that compromise device integrity by changing stored information or therapy settings. Section V extends the discussion of zero-power defenses into the realm of device design. Finally, Section VI offers concluding remarks.

## II. BACKGROUND, MODEL, AND RELATED WORK

This section summarizes the characteristics and medical usage of a modern implantable cardioverter defibrillator. It also introduces some of the equipment we used in our analyses. Following this introduction, we construct a security model that classifies potential adversaries in terms of their capabilities. Finally, we summarize previous research that motivates and informs the methods and results of this work.

### A. Implantable Cardioverter Defibrillators (ICDs)

An *implantable cardioverter defibrillator* (*ICD*) is a device that monitors and responds to heart activity. ICDs have modes for pacing, wherein the device periodically sends a small electrical stimulus to the heart, and for defibrillation, wherein the device sends a larger shock to restore normal heart rhythm. A physician surgically implants the ICD below the patient's clavicle and close to the skin (Fig. 1). The physician also implants electrical leads that connect the ICD to the heart muscle. Post-surgery, a health care practitioner can use an external *programmer* to perform diagnostics, read and write private data, and adjust therapy settings. A malfunctioning or maliciously configured ICD could harm a patient in multiple ways, including by inaction (failure to deliver treatment when necessary) or by extraneous action such as a *command shock* when the heart is beating normally.

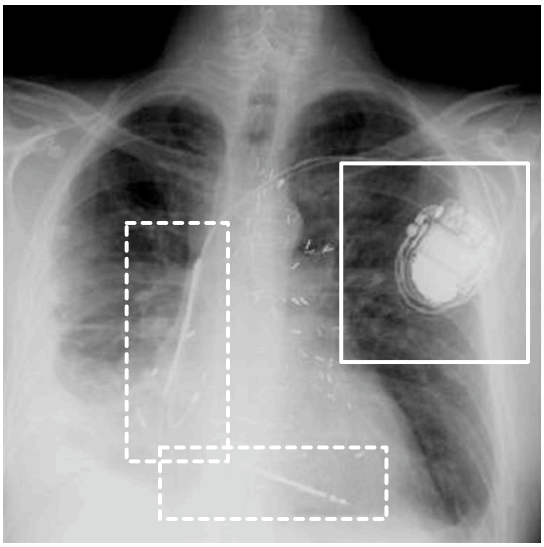

Fig. 1. Chest xray image of an implanted ICD (top right, near shoulder, solid outline) and electrical leads connected to heart chambers (center of rib cage, dotted outline).

**Magnetic switch.** Inside the ICD is a *magnetic switch*. A magnetic field in proximity to this switch causes it to close, which in turn causes the ICD to wirelessly transmit telemetry data, including electrocardiogram (EKG) readings. (We discovered, however, that we can activate transmission of telemetry on our ICD solely with an RF command and without the presence of a magnet; see Section IV.) In a clinical setting, the magnetic field comes from a magnet in the *programming head*, which is the component of the programmer that is placed in proximity to a patient's implanted ICD. At the surface of one programming head we measured this magnet at 700 gauss.

**Wireless communications.** Our ICD wirelessly communicates with the external programmer using the 175 kHz band, which is intended for short-range communications. Newer ICDs can communicate at both the 175 kHz frequency and in

the 402–405 MHz Medical Implant Communications (MICS) band [26], the latter intended for longer-range communications. One motivation for incorporating longer-range communications in new ICDs is that doing so provides greater flexibility in both clinical and home settings; for example, a patient's ICD could transmit data to an at-home monitor while the patient sleeps. (The specific communication ranges in Section I and throughout this paper are for the commercial ICD programmers we used in our experiments. We did not experiment with increasing the communications ranges of the ICDs.)

**Diversity in IMDs.** When considering implantable medical device security and privacy, it is important to draw distinctions between classes of devices that have different physical properties and healthcare goals. This paper discusses ICDs and pacemakers together because they are common devices with overlapping functions and similar operating environments.

Designers of IMDs make design decisions based in part on the fundamental properties of the problems the devices address. Some IMDs, like modern ICDs and pacemakers, are entirely self-contained with respect to power and connectivity. They are designed to last for several years, use non-rechargeable internal batteries, and have no physical connections (e.g., tubes) outside the body. Other IMDs with computational capabilities are more exposed, like cochlear implants, and are designed to last for the patient's entire lifetime. Such devices might utilize externally worn, rechargeable batteries or, like insulin pumps, might have tubes leading outside the patient. These external channels and recharging requirements could potentially make such devices susceptible to human error. While non-computational implantable devices exist, such as artificial joints, this paper considers only those IMDs that have computational capabilities.

### B. Security Model

Our research focuses on evaluating and improving the security and privacy of communication between ICDs and external ICD programmers. We consider attacks by three classes of adversaries (see also Table I):

- An adversary with a *commercial ICD programmer*, i.e., an external device commercially produced and marketed for use with ICDs. At least for the programmers with which we have experimented, there are no technological mechanisms in place to ensure that programmers can be operated only by authorized personnel.
- A *passive adversary* who eavesdrops on communications between the ICD and a commercial programmer. This adversary can record RF messages output by ICDs and programmers. This adversary might use standard or custom-built equipment, including oscilloscopes, software radios, amplifiers, and directional antennas.
- An *active adversary* who extends the passive adversary with the ability to generate arbitrary RF traffic, not necessarily conforming to the expected modulation schemes or FCC regulations. This attacker may interfere with legitimate transactions or create spurious ones by, e.g., spoofing a commercial programmer.

For the purposes of this research we assume that ICDs are honest and that they attempt to follow the protocols as specified; we do not experiment with adversarial actions that employ (possibly fake) ICDs to compromise or otherwise adversely affect the operation of commercial programmers.

### C. Related Work

Past research has investigated the challenges of manufacturing and providing safe computer-based medical treatments in the presence of unintentional failures (e.g., accidents in radiation treatments from the Therac-25 [16]). Our work from the perspective of security and privacy investigates how to provide safety and effectiveness in the presence of *intentional* failures. In the more general study of medical device security, some research focuses on securing patient data in a medical database [22]. Work by Venkatasubramanian and Gupta [31] has focused on pervasive health care security, including security involving medical sensors. Our earlier work [10] surveys a wide range of IMD security issues, including the need to balance IMD security and privacy goals with safety and effectiveness goals. In contrast, our current study is based on the systematic and pragmatic analysis of the security of a real, commercial device.

There is also a body of research studying wireless security in low-power environments, especially in the areas of sensor networks [13], [23] and wireless body area networks [32]. In contrast with these works, our zero-power security approaches eliminate the stored-energy overhead of cryptography. Chae *et al.* [2] also used RF power to implement RC5 on the WISP; we leverage their work in the context of medical devices and extend it with the new techniques of zero-power notification and zero-power human-sensible key exchange. A separate approach to our sensible key exchange uses physiological values as keys [3]. The notion of plaintext key exchange via physical contact appears in work by Stajano and Anderson [29]. Our work extends that notion by allowing key exchange to occur over an acoustic, rather than an electrical, channel.

Previous research, such as that of Goodrich *et al.* [8] and McCune *et al.* [20], considered cryptographic operations, like key agreement and authentication, that involved human action directed by sensory input. Our work sets a different goal, namely patient notification as a side-effect of a cryptographic operation, and accomplishes it through a combination of auditory and tactile feedback.

Finally, there is work using software radios to receive transmissions from commercial wireless protocols, such as BlueSniff [28] and the A5 Cracking Project [15]. Our work further demonstrates the utility of software radios by using them to help reverse-engineer and then participate in previously unknown radio protocols. Earlier work that analyzed radio transmissions of RFID-enabled credit cards [11] relied on similar reverse-engineering techniques, but the radio was built on a Gumstix embedded Linux system rather than on the general-purpose Universal Software Radio Peripheral (USRP).

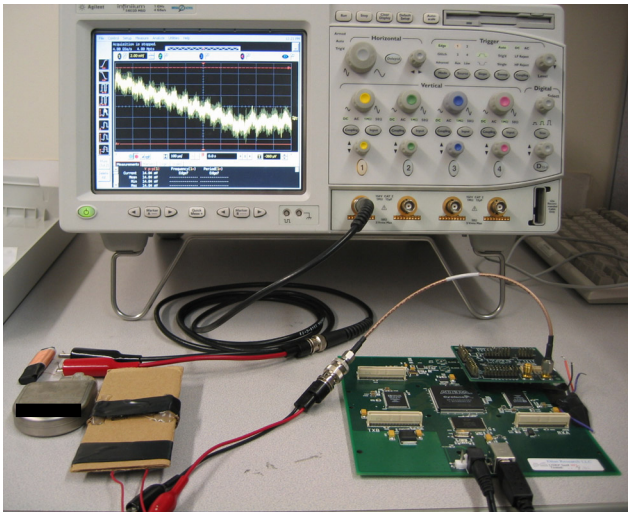

Fig. 2. Equipment used in our experiments. At top is a 4 GSa/s oscilloscope. At bottom, from left to right, are: our eavesdropping antenna, an ICD, our transmitting antenna (mounted on cardboard), and a USRP with a BasicTX card attached.

## III. Intercepting ICD Communications

We combined several reverse-engineering and eavesdropping techniques to intercept, understand, and extract information from the communications between our ICD and a commercial programmer. Our results show that wireless transmissions disclose private data.

We used two hardware tools to intercept the radio frequency (RF) signals emitted by the ICD and the programmer: a recording oscilloscope and a *Universal Software Radio Peripheral* (*USRP*) [5]. The oscilloscope is standard lab equipment; the USRP is a programmable device that interacts with open-source GNU Radio [30] libraries on a host PC. Section III-D1 describes the equipment in more detail and Fig. 2 shows a picture.

### A. Reverse-Engineering Transmissions

We began by capturing RF transmissions around 175 kHz. Using an oscilloscope, we were trivially able to identify transmissions from our ICD and the commercial ICD programmer. We saved traces from both the oscilloscope and the USRP. We processed these RF traces in software (using Matlab and the GNU Radio toolchain) to recover symbols, then bits. Finally, by analyzing these bits we discovered key aspects of the ICD's protocols and the data that it and the programmer transmit.

**The physical layer.** Before we could analyze protocols at the application layer, we needed to determine the data bits that corresponded to the raw electromagnetic signals in the traces we obtained with the oscilloscope and USRP. For completeness, Section III-D2 discusses radio terminology and describes the process of extracting bits from RF traces. We determined that the ICD and the programmer share an encoding scheme but use different modulation schemes. Fig. 3 shows segments of the transmissions we examined.

**Transmissions from the programmer.** In reverse-engineering the programmer's transmissions, we had an advantage: a serial connection between the programmer device and the programming head carries the raw bits to be transmitted. By tapping this serial connection we were able to obtain these bits for comparison with the encoded and modulated RF signals output by the programmer's radio.

Through spectral analysis of the programmer's RF transmissions, we determined that it uses binary frequency shift keying (2-FSK) as its modulation scheme. We confirmed this by demodulating bits from the RF trace and comparing the results to the raw bits we collected on the serial line; we found them to be identical. We also determined via standard techniques that the length of a single symbol transmitted by the programmer is 14 cycles of the center frequency, making the symbol rate 12.5 kBd (i.e., 12 500 symbols per second).

**Transmissions from the ICD.** Reverse-engineering the ICD's transmissions was more difficult because we did not have access to a wire carrying raw bits. However, we knew that the ICD transmits certain stored information, so we inserted information in the ICD using the programmer (by, for example, setting the patient name to a string of 'A's). We analyzed the RF signal to identify phase shift-keyed bits and, using our cribbed patient name, learned that the ICD uses a modulation scheme known as differential binary phase shift keying (DBPSK). We also determined that the symbol length of ICD transmissions is two cycles of the carrier wave, making the symbol rate 87.5 kBd.

**Decoding.** When we attempted to decode the demodulated symbols, we looked for the cribs (known plaintexts) we had inserted. We observed that transmissions from both ICD and programmer are encoded under Non-Return-to-Zero Inverted (NRZI) with bit stuffing. Section III-D2 explains this encoding scheme and Fig. 5 shows an example of NRZI decoding.

### B. Eavesdropping with a Commodity Software Radio

We built an eavesdropper using the Universal Software Radio Peripheral (USRP) in concert with the open source GNU Radio libraries. For the initial analysis in Section III-A, we simply used programs included with GNU Radio to capture and store received radio signals, then wrote code in Matlab and Perl to analyze those signals. To eavesdrop in real time, we integrated the necessary functions back into the C++ and Python framework of GNU Radio. This section describes the eavesdropping process in detail and shows the results of our passive attacks.

**Establishing a transaction timeline.** Our first step toward understanding where and when to eavesdrop was to establish a timeline for bidirectional conversations between the ICD and the programmer; this timeline is shown in Fig. 4. We established the timeline by interacting with the programmer and capturing programmer and ICD transmissions on an oscilloscope. We did not need to decipher these transmissions; we were able to infer their meanings and some of their contents by observing the order in which the programmer acquired

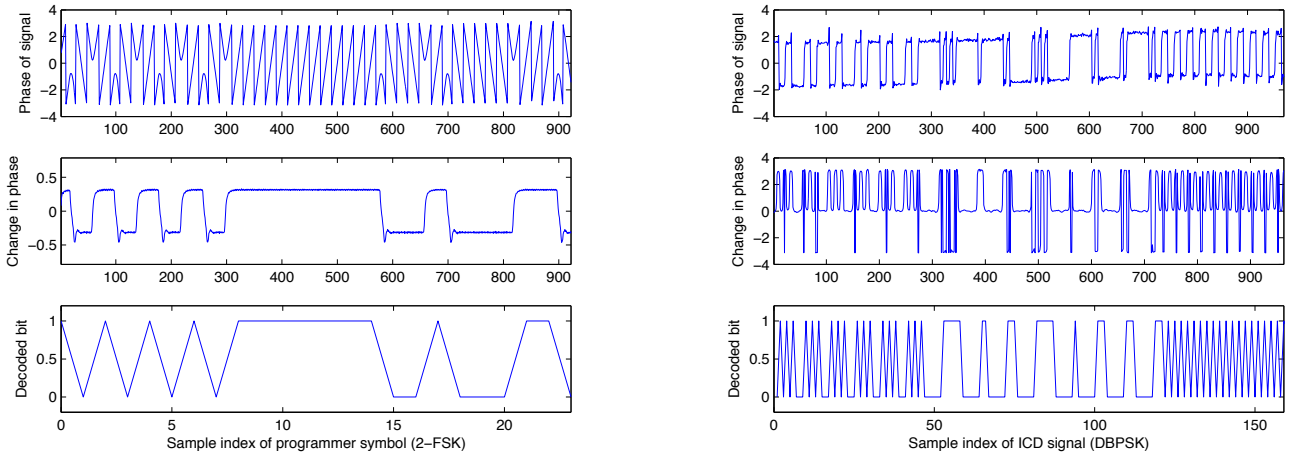


Fig. 3. Demodulating received programmer (left) and ICD transmissions. The top plot in each figure shows the phase of the raw RF signal, downconverted from 175 kHz to baseband. Both 2-FSK and DBPSK encode data by the phase *change* of the signal, pictured in the middle row. The final row shows the decoded bits: in 2-FSK the bit is determined by the sign of the phase change, and in DBPSK by whether it is closer to 0 or $\pi$. Note that there are fewer bits than samples; our 500 kHz sampling rate generates 40 samples per programmer symbol and about 6 per ICD symbol.

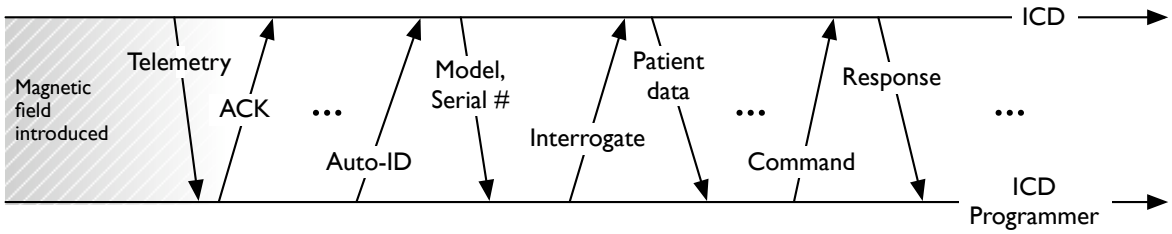


Fig. 4. Timeline of a conversation between an ICD programmer and an ICD. If a programmer is present it will acknowledge each packet automatically. When told by an operator to do so, the programmer asks the ICD for identifying information, which the ICD provides. The programmer then interrogates the ICD for patient data, which the ICD provides. Other commands (such as ICD programming commands) and their responses follow.

information about the ICD. We used the timeline to determine which transmissions to inspect using GNU Radio and our own tools.

**Using GNU Radio.** One builds a GNU Radio program by assembling digital signal processing *blocks* into an information *flow graph* that connects a *source* to a *sink*. If a suitable hardware device such as a USRP is attached, source and sink blocks can represent radio interface cards. Intermediate blocks perform signal and data processing steps. Because transmissions from ICDs and ICD programmers differ greatly in amplitude, because their modulation schemes differ, and because we wanted to assess our ability to eavesdrop on the two sides separately, we developed a slightly different GNU Radio receiver for each end. See Section III-D3 for more details regarding one of those receivers.

An example illustrates the relative ease with which one can develop a GNU Radio block: while inspecting demodulated and decoded traces in search of patterns, we discovered byte sequences that appeared to be packet delimiters (`1000000` for end-of-packet and `1111110000001` for beginning-of-packet). We built a GNU Radio block to packetize bitstreams and gather data about the resulting packets. This block is logically simple and was adapted from similar functionality in the GNU Radio library. The complexity of this change was modest: we removed 87 of 166 total lines of C++ code (discounting comments and whitespace) from `gr.correlate_access_code_bb` and `gr.framer_sink_1`, and added 44 lines of code. For perspective, the C++ source files for the other blocks used in this receiver contain roughly 1600 lines of code, mostly implementing signal processing operations.

**Intercepting Patient Data.** Analysis of our captures via the reverse engineering process described in Section III-A revealed several things. First, we were able to find cleartext representations of a wide range of what would have been patient data in captured transmissions. (We experimented with artificial patient data that we stored on the ICD; we did not experiment with real patient data.) Even without knowledge of the semantics of the packet format, these data are easily extractable.

From this we conclude that this model of ICD programmer and this model of ICD do not protect their transmissions cryp-

tographically against disclosure when communicating with each other. The personal data transmitted in cleartext include the patient's name, date of birth, medical ID number[2], and patient history. Equally easy to find are the name and phone number of the treating physician, the dates of ICD and lead implantation (which may differ), the model, and the serial number of the ICD and leads. This list is not exhaustive; we observed other items of personally identifying data being transmitted in cleartext. All this information is sent either in the clear or in inverted form from the ICD to the programmer during interrogation. Furthermore, for the fields we manipulated via reprogramming attempts, these fields are sent in the clear from the programmer to the ICD.

**Intercepting Telemetry.** Our ICD begins to broadcast telemetry data in cleartext as soon as a magnetic field of sufficient strength is introduced. We activated telemetry with a magnet we measured at 700 gauss by placing it within 5 cm of the ICD. Telemetry transmissions, sent over the 175 kHz FM band at a rate of 32 packets per second in our experiments, contain representations of a patient's electrocardiogram (EKG) readings; these readings convey heart rate and other potentially private information about the patient's cardiac activity in real time. We determined this information leak with a known-plaintext attack on ICD telemetry in which we attached a function generator to one of the ICD's sensing ports with the voltage set larger than that generated by a real heart. Feeding the device a predictable signal (in place of a heartbeat), we were then able to observe the same period in the payloads of the ICD telemetry packets. Varying the period of the input resulted in a change of the output frequency to match, and in the absence of our input the telemetry data appeared as aperiodic binary noise (values from 0x00 to 0x03).

### C. Limitations

Because our goal was to assess disclosure of private data, we did not fully reverse-engineer the communication protocol; rather, we inspected captured traffic and learned where to look for certain cribs. In particular, we were not always able to predict the value of a certain field which we believed to be a checksum. Another field appeared to be a sequence number that increased according to a rule we did not fully investigate.

As evidenced by the ICD programmer's many (more than 10 screens) informational displays and data graphs, the ICD transmits a large amount of information to the programmer. We looked for cribs that we expected to see encoded under fairly predictable schemes (e.g., ASCII). We did not attempt to map all of the fields shown in the programmer's menus to their counterparts in the radio transmissions.

We do not provide an estimate of the upper bound on communication distance, as this distance is sensitive to many factors that are beyond the scope of this paper. Without an amplifier or any other specialized hardware, we were able to eavesdrop on both the ICD and the ICD programmer from a distance of several centimeters.

[2]A medical ID number need not be globally unique; it is assigned according to the policy of the hospital and may, in some cases, serve as an index into a database of patient records.

### D. Auxiliary Information

This section provides supplementary information about our equipment, our reverse engineering of the physical layer, and our programming of software radios. The reader can view this section as the appendix to Section III. Skipping this section should not affect one's understanding of the core contents of this paper.

*1) Equipment:* Capturing RF packets requires equipment capable of operation at the relevant frequency bands — in our case, an antenna tuned to 175 kHz. We achieved our best results with an antenna that we removed from an explanted pacemaker and an antenna that we removed from a Medtronic Carelink device (model 2490C). We also performed successful eavesdropping experiments using a homemade, hand-wound, ferrite core antenna and even a simple closed loop made by joining the grabbers of an oscilloscope probe. We used the same antennas for our replay attacks, sometimes in combination with an amplifier; once again, we achieved our best results with the antennas removed from the pacemaker and the Carelink device.

The oscilloscope eavesdropping setup is trivial: the oscilloscope samples the voltage on an attached eavesdropping antenna at regular time intervals, and we store these $\langle$*Time, Voltage*$\rangle$ pairs to disk for later offline processing. The oscilloscope proved very useful in our research; however, its limited memory meant that we could record only about eight seconds of RF communication at a time, which was insufficient for recording an entire conversation between our ICD and programmer.

Because of the oscilloscope's limited memory depth, we also used a *Universal Software Radio Peripheral* (*USRP*) [5] in our analysis. The USRP is a single board containing an FPGA for fast signal processing and swappable radio interface cards called daughterboards. We used a *BasicRX* daughterboard, suitable for low-frequency communication, to interface with our eavesdropping antenna. The USRP records signals as complex I/Q samples, which are interconvertible with the data format used by the oscilloscope. The daughterboards and FPGA perform RF conversion in hardware and stream the results to the PC. The USRP can sample at a rate of up to 8 MHz (32 MB/s) over a time range limited only by hard drive capacity; we performed the majority of our experiments using this setup with a sampling rate of 500 kHz.

*2) Reverse-Engineering the Physical Layer:* Two key processes are used in transmission of data over RF: encoding and modulation. Encoding is the process used to convert data bits into radio symbols, and modulation determines how the hardware varies the carrier radio wave to transmit those symbols. In order to receive data, a receiver must first demodulate and then decode using the respective schemes. In RF systems with asymmetric hardware constraints, such as with ICDs and programmers, it is not uncommon for transmissions from different devices to use different modulations and encodings.

We found that this is indeed the case for the programmer and the ICD.

To collect RF data, we connected one of our antennas to a recording device (oscilloscope or USRP), then placed the antenna in close proximity to the ICD and programmer as they communicated wirelessly. We determined that the modulation scheme used by the programmer is binary frequency shift keying (2-FSK). In 2-FSK, the programmer encodes symbols by transmitting at a different frequency for each symbol state. In this case, 150 kHz and 200 kHz respectively represent the two possible states.

The ICD uses a different modulation scheme, differential binary phase shift keying (DBPSK), in which the two possible symbols are represented by transmission at the same frequency but opposite phase. DBPSK is a common modulation scheme for resource-constrained devices because it remains robust when the transmitting oscillator is inaccurate. Such oscillator variability is often observed in low-power devices that save energy by partially or fully disabling RF circuitry when it is not needed. One technique a device like an ICD may employ is to turn off the synchronization circuitry that keeps the RF carrier frequency accurate between transmissions. We observed some evidence of this method in the ICD we studied.

Fig. 5 shows the symbols we obtained by demodulating part of a packet from a programmer transmission in which the programmer issues commands to the ICD to change the stored patient name to `AA AAAA`. As can be seen in row (a), different patterns of symbols can be used to express the same pattern of bits. For example, `111111011` and `000000100` can both correspond to ASCII '`A`'. In determining the encoding scheme, we experimented with several common binary line code schemes (e.g., Manchester). By searching for the known plaintext crib of the patient name, we ultimately determined that both programmer and ICD transmissions are encoded in NRZI (Non-Return-to-Zero Inverted) form with *bit stuffing*. In NRZI encoding, zero bits are represented by no change in symbol over one symbol period, and one bits are represented by a change of symbol state. Bit stuffing is a common technique (used in the USB standard, for example) in which data containing the end-of-frame delimiter (EFD) is broken up by the insertion of an extra bit to allow their transmission without prematurely ending the frame.

*3) Programming the Software Radio:* Fig. 6 gives the block diagram of our eavesdropper for ICD programmer transmissions. Complex samples are streamed from the USRP with BasicRX daughterboard to the PC via USB, then output by the `gr.usrp_source_c` block. The frequency demodulation block `gr.blks.fm_demod_cf` computes the frequency shift of the incoming signal and finds the correct sampling interval for each signal. These symbols, in the form of frequency shifts, are mapped to bits by `gr.binary_slicer_fb`, and the bits are NRZI-decoded using `gr.diff_decode_bb`. This functionality is implemented entirely with existing blocks built into the GNU Radio library. Finally, the resultant bits are framed as packets by our `imd.sink_175` block, which we adapted from code in the GNU Radio library.

For active attacks, as for passive attacks, we employ several of GNU Radio's standard signal processing blocks. A `gr.file_source` block streams complex samples from a file on disk. These are processed by an `imd_clean_fmmod` block of our own design (80 lines of C++) which separates programmer transmissions from background noise and ICD transmissions, then generates a clean FM-modulated signal carrying the exact same data. The clean signal is amplified in software and then streamed into `usrp.sink_c`, which passes the signal to the USRP for transmission over the air.

## IV. Active Attacks with a Commodity Software Radio

We implemented several active attacks using the USRP and a *BasicTX* daughterboard to transmit on the 175 kHz band. Because the BasicTX card lacks built-in amplification, we also interposed a simple RF amplifier circuit for many of our replay attempts. With amplification, we were able to mount active attacks across an air gap of several centimeters; we did not attempt further amplification or longer distances. Without amplification, we successfully mounted selected replay attacks with the antenna from a Carelink device within one cm of the ICD. Additionally, during the course of our experiments, the ICD entered its *elective replacement indicator* (ERI) mode, which normally indicates to physicians that ICD replacement should be scheduled. We successfully mounted all of our replay attacks before and after the ICD entered this mode.

All of our active attacks fall into the category of replay attacks: only simple waveform manipulation and repetition — not packet analysis or reassembly — were necessary. Creating a software radio programmer capable of emulating all the functions of a commercial programmer would require additional reverse engineering; we chose instead to focus on attacks that are both significant and illuminating (in particular, those that violate confidentiality, integrity, or availability) and that pose minimal complexity to adversaries.

### A. Replay Attacks

For simplicity, our replay attack technique is *transmit-only*: we did not attempt to synchronize replayed programmer packets with the ICD's response packets. The penalty we paid as a result was that each successful replay attack was preceded by zero or more unsuccessful attempts. It was generally sufficient to set the ICD to a known state, replay the desired transmission in a loop for several minutes, and then re-evaluate the resulting state of the ICD. This process allowed us to determine whether one or more replays were successful but did not reveal the precise number of successes. The transmissions we replayed varied in length from one second to 37.7 seconds. We did not attempt to optimize our attacks by removing silent periods from the replayed traces. Each of our replay attacks targeted only the ICD against which we recorded the original trace.

In order to rule out the possibility that proximity of the magnet in the programming head is necessary for the ICD to accept programming commands, we tested each replay attack with and without a magnet near the ICD. In all cases, both

```
(a)    101010100000001|...|000000100|111111011|000000100|111111011|...|1111111
(b)    _11111110000001|...|100000x10|100000x10|100000x10|100000x10|...|1000000
(c)    ------SFD------             A         A         A         A     --EFD--
```

Fig. 5. Part of a sample programmer transmission containing the crib text `AAAA`. Row (a) represents the demodulated bits, (b) are the NRZI-decoded bits, and (c) are the bits rendered as ASCII. The Start-of-Frame and End-of-Frame Delimiters pictured mark the beginning and end of each packet.

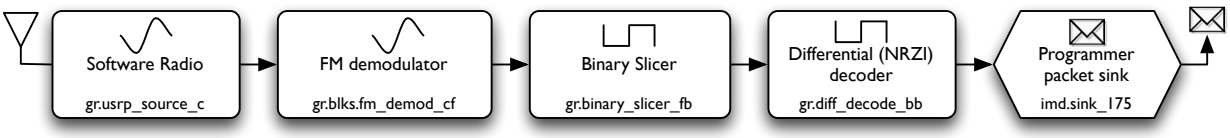


Fig. 6. Architecture of a GNU Radio-based eavesdropper. The purpose and Python class name of each signal processing block is included, as well as the format of the data it processes. All blocks but the last are from GNU Radio's built-in block library. We adapted the last block, our `imd.sink_175` packet framer, from built-in GNU Radio blocks.

scenarios were successful. We conducted all of our active attacks with our commercial programmer turned off.

**Triggering ICD identification.** We performed several experiments in which we replayed a 1.5-second *auto-identification* trace we had recorded from the programmer. Each of these transmissions resulted in an identical response from the ICD, disclosing the ICD's presence and several details about the device such as its model and serial number. As the auto-identification command is the first set of packets sent by a programmer in a normal session, our experiments suggest that no prior synchronization is required for a successful exchange as long as the programmer transmits during ICD radio silence.

**Disclosing patient data.** After the auto-identification step, the programmer asks the ICD for the rest of the information stored on it, including patient data. We used GNU Radio to replay a 26-second capture containing both an auto-identification command and the *interrogation* command that elicits the more detailed data. Using the demodulation and framing code described in Section III-A, we confirmed that the ICD responded to our interrogation command with the same response it gave to the ICD programmer's original command. This response includes personal information such as patient name, diagnosis, and many other details.

**Disclosing cardiac data.** We observed that certain conditions cause the ICD to send periodic telemetry transmissions at a rate of 32 packets per second. If a sufficiently strong magnet is near the ICD, the ICD appears to transmit telemetry indefinitely. When the strong magnet is taken away (as when the programming head is removed from the patient's skin), these transmissions stop after 10 seconds. However, introducing a magnet is not the only way to elicit telemetry transmissions from the ICD. We observed that replaying certain sections of recorded conversations, in particular the beginning of the interrogation command sequence, caused the ICD to emit packets for several seconds after the end of the replay. We hypothesize that an attacker could replay these commands in a tight loop to elicit continual telemetry from the ICD.

**Changing patient name.** Medical personnel can learn a patient's name by interrogating an ICD. We used GNU Radio to replay traces in which the programmer changes the patient name stored on the ICD. After ten (on average) replays of the same trace, more than one of which may have succeeded, we used the programmer to confirm that we had successfully changed the patient name stored on the ICD. We repeated this experiment several times, each time changing the name to a different value and confirming the change on the programmer.

**Setting the ICD's clock.** The ICD programmer and the ICD have separate clocks. The ICD's clock allows it to record timestamps in its event log and can be set from a menu on the programmer. We used GNU Radio to replay traces in which the programmer sets the time or date on the ICD. We then confirmed in a new programming session that we had successfully set the ICD's clock. As with the patient name change, this attack succeeded after an average of ten replays, more than one of which may have succeeded.

**Changing therapies.** *Therapies* are the ICD's responses to cardiac events. A commercial programmer can be used to enable and personalize therapies for an individual patient's medical needs or to disable (i.e., turn off) the device's life-saving functions. We used GNU Radio to replay captures in which the programmer turns off therapies. With therapies turned off, the ICD does not respond to potentially dangerous cardiac conditions. After 24 replay attempts, more than one of which may have succeeded, we confirmed in a new programming session that we had successfully disabled all of the therapies we had enabled before our attempts.

**Inducing fibrillation.** During implantation surgery, it is common for a physician to test the newly implanted ICD to ensure that it can both sense and appropriately treat a cardiac condition known as ventricular fibrillation (V-Fib), one of the most common kinds of heart rhythm problems. Accordingly, the ICD has several testing modes in which it can induce V-Fib. Such a test — called an *electrophysiological (EP) study* — is normally conducted with cardiologists standing by to stop the fibrillation if the ICD fails to do so. After a physician puts the programmer in EP study mode, sets certain study parameters, and explicitly confirms that the study should begin, the programmer sends the ICD a sequence of commands that

requests a low-energy ($\sim$1 joule) *command shock* to be applied to the patient's heart at a precise point in the patient's cardiac rhythm, with the goal of inducing V-Fib. When its automatic therapies are enabled, the ICD should immediately detect and treat the fibrillation by delivering the proper therapy.

We introduced a 100 $\Omega$ resistor between two of the ICD's labeled defibrillation ports to measure the voltage applied during a command shock. We then used our commercial programmer to conduct an EP study in which we sent a 1.0 J shock from one defibrillation port to the other across the resistor. Using our oscilloscope, we measured the pulse's peak voltage at an average of 138.4 V over three trials. We then replayed a recording of the EP study command sequence via our software radio. At least three of 30 replay attempts succeeded in causing similar voltage spikes, averaging 137.7 V. Besides observing voltage spikes on the oscilloscope, we confirmed that the ICD's *last high-voltage therapy* field, shown in a programmer menu, changed to reflect the date and time of our last successful attack. We successfully triggered command shocks via replayed commands even after turning off all of the ICD's automatic therapies.

The commercial programmer's user interface provides safeguards to make it difficult for a physician to accidentally issue a command shock when the ICD's therapies are disabled. Our successful replay attacks demonstrate that although these safeguards are implemented in the programmer's software, an adversary who bypasses the commercial programmer using a software radio could circumvent these safeguards. The broader lesson is that external devices such as commercial programmers should not be considered part of an IMD's trusted computing base. Additionally, we argue that if any IMD exhibits a *test procedure* $T$ for some *property* $P$, and if there are no medical reasons for conducting procedure $T$ other than testing property $P$, then it should be *impossible* to trigger $T$ unless $P$ is enabled. For example, as our experiments suggest, if $P$ is the efficacy of the device when therapies are enabled and $T$ is a test, then the ICD — not only the external programmer — should verify that therapies are enabled prior to conducting the test $T$.

**Power denial of service attack.** Our experiments suggest that the ICD could be forced to remain in a mode in which it continually engages in wireless communications. As we discuss above, a strong magnetic field causes the ICD to transmit telemetry continually, and the ICD responds to RF commands without the presence of a nearby magnet. We have not measured the power consumed by telemetry or other RF transmissions, but it is possible that these operations decrease battery life faster than normal ICD operation alone.

**Other attack vectors.** As noted in our earlier work [1], [10], there may be other attack vectors against IMDs, such as insecure software updates or buffer overflow vulnerabilities. We do not experiment with such attack vectors in this work, but note that the existence of such exploitable vulnerabilities could allow for further adversarial control over the state and operation of an IMD.

## V. Zero-Power and Sensible Defenses for IMD Security and Privacy

Providing security and privacy on an IMD involves health risk factors and tight resource constraints. Traditional approaches could potentially introduce new hazards to patient safety. For instance, protecting an IMD with a cryptographic key may provide security, but the unavailability of a key could hinder treatment in emergency situations. Another risk to IMD availability is excessive power consumption by mechanisms other than those needed for the device's primary function. For instance, the energy cost of performing computation for cryptography or radio communication could directly compete with the energy demands of pacing and defibrillation. Effective mechanisms for security and privacy should not provide new avenues for an unauthorized person to drain a device's battery. For instance, spurious wake-ups or a cryptographic authentication process itself could cause a device to enter a state that consumes excessive amounts of energy (as in, e.g., the *sleep deprivation torture* attacks of Stajano and Anderson [29]).

Therefore, three goals guided our design of zero-power approaches for IMD security and privacy. First, an effective approach should either prevent or deter attacks by both malicious outsiders with custom equipment and insiders with commercial programmers. Because IMD therapies rely on long-lasting batteries, a second goal is that security and privacy should draw no power from the primary battery, thus preventing denial of service attacks on power. Third, security-sensitive events should be effortlessly detectable by the patient. We must also ensure that new security mechanisms do not introduce new failure modes.

Our contributions include three *zero-power defenses* and prototype implementations, one of which we evaluated for effectiveness in a substance approximating the radio properties of human tissue. *Zero-power notification* harvests induced RF energy to wirelessly power a piezo-element that audibly alerts the patient of security-sensitive events at no cost to the battery. *Zero-power authentication* similarly harvests RF energy to power a cryptographically strong protocol that authenticates requests from an external device programmer. Finally, *sensible key exchange* combines techniques from both zero-power notification and zero-power authentication for vibration-based key distribution that a patient can sense through audible and tactile feedback. While we implemented prototypes of our proposed defenses, we did not incorporate our prototypes into a real IMD. (We use the term *zero-power* only to emphasize that no expenditure of energy from the primary battery is necessary. Zero-power defenses are also a step beyond the use of a secondary battery for security-only or other auxiliary purposes.)

We do not claim that our defenses are final designs that IMD manufacturers should immediately incorporate into commercial IMDs. Rather, we believe that our research establishes a potential foundation upon which others can create, evaluate, and implement new defensive mechanisms for future IMD designs.

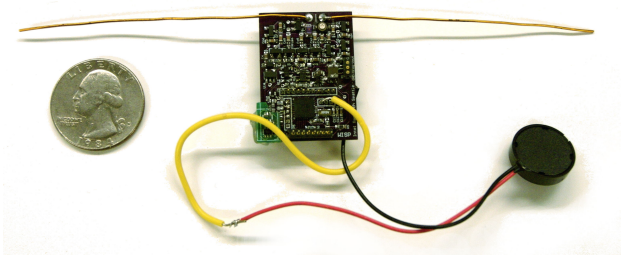

Fig. 7. The WISP with attached piezo-element.

### A. Detection: Zero-Power Notification for Patients

As our earlier work notes [10], it may be possible to deter malicious activities by making patients aware of those activities. Our zero-power notification alerts a patient to potentially malicious activities both by insiders using commercial programmers and by outsiders using custom attack hardware, thereby making patients effortlessly aware of remote communications. On some modern ICDs, triggering the magnetic switch causes the ICD to beep. Whether intentional or not, such beeping represents a step towards the concept of patient awareness by way of audible alerts. But beeps triggered by a magnet alone do not raise patient awareness for RF-initiated actions, which our approach does.

**Our approach: WISPer.** Our prototype of zero-power notification wirelessly drives a piezo-element that can audibly warn a patient of security-sensitive events. The prototype builds upon revision 1.0 of the Wireless Identification and Sensing Platform (WISP) [27], a postage stamp-sized embedded system that contains RFID circuitry and a Texas Instruments MSP430F1232 microcontroller with 256 bytes of RAM and 8 KBytes of flash memory. The WISP harvests energy from a 915 MHz RF signal generated by the Alien ALR-9640 nanoscanner, a UHF RFID reader running the EPC Class 1 Gen 1 protocol. Although we prototyped at 915 MHz, it may be possible to create similar hardware that operates at the frequency of current ICD programmers. WISPer adds to the WISP's base code a 30-line C program that activates a piezo-element which we attached to the general-purpose I/O (GPIO) ports of the WISP. After WISPer receives a sequence of wireless requests from the RFID reader, it emits constant chirping, thereby informing the patient of the wireless interaction. A future version of WISPer could set a separate GPIO high after buzzing for a certain number of cycles, and the IMD could allow remote communications only after that GPIO is raised. WISPer satisfies our zero-power notification design constraints: it draws no energy from a battery and can issue alerts for all reprogramming activity.

**Evaluation.** Two measurements quantify the effectiveness of the WISPer prototype for zero-power notification. We used a sound level meter to measure Sound Pressure Level (SPL) with a reference pressure of 20 micropascals (the standard for above-water calculations). The buzzing volume peaked at 67 dB SPL from a distance of 1 m. For reference, a normal conversation is about 60 dB SPL and a vacuum cleaner at a distance of 3 meters is about 70 dB SPL [17]. We then placed our prototype in an environment designed to simulate implantation in a human (Fig. 8). We implanted the device beneath 1 cm (a standard ICD implantation depth) of bacon, with 4 cm of 85% lean ground beef packed underneath. We took several readings at the surface of tissue in order to ascertain the effects of obstruction by tissue. We measured 84 dB SPL of sound at the surface of the tissue, and subjectively were easily able to hear it from a meter away (more than the distance between standard ICD implantation sites and a patient's ear).

These tests of our prototype device suggest that its piezo-element is audible under reasonable simulations. Because malicious attackers may attempt their attacks in noisy, chaotic environments to vitiate auditory notification, and because some patients with ICDs may have limited hearing, we note that a piezo-element can be used to produce vibration instead of audible sound. In our experiments, the 4 kHz alert used was easily sensed by touch.

### B. Prevention: Zero-Power Authentication

Our second defense implements a zero-power method that allows an IMD to verify that it is communicating with a real commercial programmer (and not an unauthorized software radio programmer).

**Approach.** The device implements a simple challenge-response protocol (Fig. 10) based on RC5-32/12/16 [24]. In this model, all commercial programmers know a master key $K_M$, each IMD has an serial number or identity $I$, and each IMD has an IMD-specific key $K = f(K_M, I)$, where $f$ is any cryptographically strong pseudorandom function (such as AES). The value $K_M$ should be stored in secure hardware on the programmers. The protocol works as follows. The programmer transmits a request to authenticate to the WISP. The WISP responds with its identity $I$ and a nonce $N$. The programmer computes $K = f(K_M, I)$ to get the IMD-specific key and then returns the response $R = \text{RC5}(K, N)$ to the WISP. The WISP computes the same value and verifies the value it received from the programmer against its result. The WISP finally sets a GPIO high which, if attached to or built into a real IMD, would inform the IMD that the WISP successfully authenticated a programmer.

For the sake of simplicity, our prototype does not implement the full protocol. Namely, in our experiments we use a fixed nonce and assume that the programmer knows the nonce in advance. Using this simplified model, we experimentally verified that, upon receiving the programmer response $R$, the WISP was able to perform its own RC5 encryption and verify equality. We were able to run this subset of the protocol with complete reliability using only harvested energy. To lift from the simplified model to a real implementation of our protocol, we note that the nonce should appear random to an adversary. Since we, and others [2], show that it is possible to run RC5 on a WISP, a natural solution would be to generate the nonce with RC5 in counter mode. A better approach that would yield a truly random nonce is to exploit process variations

Fig. 8. To simulate implantation in a human, we placed the WISP in a bag containing bacon and ground beef.

and omnipresent thermal noise by extracting random bits from SRAM using the FERNS technique of Holcomb *et al.* [12]. Applying FERNS to 256 bytes of SRAM could yield 100 bits of true randomness each time the SRAM is powered up. Our work would benefit from an implementation of the memory-as-TRNG technique on the WISP.

**Evaluation.** We learned from our successful attacks that private data transmitted between our ICD and programmer are not encrypted. We propose that cryptography be added at least at critical junctures. Encryption of the entire conversation would be optimal — for example, a secure channel between programmer and ICD could prevent third-party disclosure, replay, and many other attacks — but in the interest of modularity we consider in this paper only defensive approaches that might be implemented with less extensive modifications to current ICD designs. Modularity aside, if we were to propose cryptographic extensions that required significant changes to ICD design, it would be necessary to consider the power cost of our proposed changes. Without detailed knowledge of the inner workings of ICDs, however, we cannot accurately assess the cost of adding cryptography to existing devices.

The tension between increased security and increased power consumption can be resolved by requiring successful zero-power authentication *before* the device switches to higher power consumption modes. Our prototype shows that this proposal is feasible for bootstrapping stronger (and possibly mutual) authentication methods. Our prototype harvests power from RF transmissions, performs a cryptographic authentication, and on successful authentication of a programmer, sets a GPIO high which, if connected to or built into a real ICD, would permit the ICD to participate in active RF communication and other higher-level protocols. This approach addresses the risk of *sleep deprivation torture* described by Stajano and Anderson [29].

**Key management.** This paper does not address the well-known problem of key management. Using a shared secret (called $K_m$ above) is reasonable for a prototype implementation, but a large-scale deployment of shared key material — in implanted devices, hospitals, clinics, ambulances, IMD programmers, and so on — may pose an unacceptable risk because of the ease with which an unauthorized party could decrypt transmissions upon obtaining the key material. (Though our recommendation of storing $K_m$ in secure hardware does partially mitigate this risk under certain threat models.) The simple scheme described above also fails to address revocation of privilege and is therefore ill-suited to situations in which key material might be compromised, although the proposed system is still no less secure than the open-access model of conventional systems. An SKEYS [9] or key-regression [6] approach, with periodic updates of programmer keys, might mitigate the time-window in which an attacker can use compromised keys while also not significantly changing the overall model. Furthermore, the offline nature of the transactions that must be secured — imagine an ambulance reaching an ICD patient in a remote setting — further complicates the problem of key management and revocation.

In the context of medical devices, security-related design choices must balance security, privacy, safety, and efficacy [10]. An ideal key management scheme for this context, which we present as an important open problem, must provide security and support privacy without hindering the operation of medical devices that are already known to provide safe and effective treatments.

### C. Zero-Power Sensible Key Exchange

We now present a key-distribution technique that complements both of our previous defensive techniques: distribution of a symmetric cryptographic key over a human-perceptible sensory channel. The primary goal is to allow the patient to detect a key exchange while it occurs.

**Approach.** The programmer initiates our protocol by supplying an unmodulated RF carrier signal that could power the passive component of the IMD. The IMD then generates a random value to be used as a session key and broadcasts it as a modulated sound wave. The amplitude of this sound wave is such that it can be easily received and demodulated by a reader with a microphone in contact with the patient's body near the implantation site, but it cannot be heard over background noise at any appreciable distance from the patient, at least not without dedicated sensing equipment. The close proximity this enforces further ensures patient awareness and consent to the authentication attempt. Once key exchange has been performed, RF communication can safely occur over a longer range without fear of eavesdropping.

**Evaluation.** We implemented our key exchange mechanism on the WISP using as carrier frequency the same 4 kHz

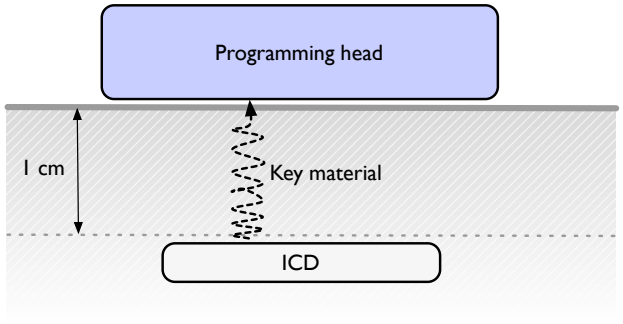


Fig. 9. Zero-power sensible key exchange: a nonce is transmitted from the ICD to the programmer using acoustic waves. It can be clearly picked up only if the programmer is in contact with the patient's body near the implantation site, and can be used as the secret key in the authentication protocol from the previous section. (1 cm is a typical implantation depth. Diagram is not to scale.)

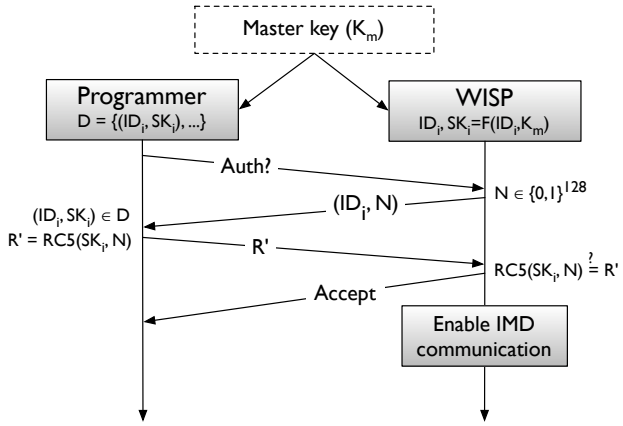


Fig. 10. The protocol for communication between an ICD programmer and a zero-power authentication device (a WISP RFID tag, in the case of our prototype).

audible and tactile signal discussed above. To effect key exchange, we used the same modulation scheme currently in use by our reader (2-FSK). We achieved a baud rate of 310 Bd, permitting transmission of a 128-bit nonce in 0.4 s. The components performed key exchange without drawing power from a battery, and the exchange was clearly audible, measuring 75 dB SPL through a human hand. When the microphone was not in contact with the skin, the sound pickup was too low to be measured on our meter ($<$ 60 dB SPL). In our *ad hoc* experiments, transmission of the key was easy to feel with the hand, but difficult to hear at a distance. While these preliminary measurements show the plausibility of making eavesdropping difficult, further work is necessary to illuminate the relationship between sound levels and the ability to eavesdrop. Furthermore, an adversary may attempt to eavesdrop on the electromagnetic emanations [14] of the electrical components that generate the sound rather than on the sound itself. Radio shielding in the form of a Faraday cage or use of non-electromagnetic, optical links between security-sensitive modules may help to reduce these unintended emanations. An alternate approach for sensible key exchange might be for the programmer to transmit the key to the IMD over an audio channel, or for the final key to be derived from keys sent in both directions.

## VI. Conclusion and Future Work

Our investigation shows that an implantable cardioverter defibrillator (1) is potentially susceptible to malicious attacks that violate the privacy of patient information and medical telemetry, and (2) may experience malicious alteration to the integrity of information or state, including patient data and therapy settings for when and how shocks are administered. Moreover, standard approaches for security and access control may not always be suitable for IMDs due to tensions between security (e.g., access for pre-authorized parties only) and safety (e.g., access for previously unauthorized parties in emergency circumstances) [10]. Our three new methods for zero-power security (zero-power notification, zero-power authentication, and sensible key exchange) implemented on a prototype are steps toward mitigating this tension without simultaneously drawing power from a battery.

**Reflections on existing and next-generation IMD technologies.** Evaluating the security and privacy of an IMD best leverages skills from many disciplines, including security, cryptography, cardiology, signal processing, radio communications, and antenna design. Next-generation IMDs, which may incorporate greater communications capabilities and be more networked, should not rely solely upon external mechanisms like firewalls on external devices and controlled distribution of commercial programmers. Firewalls on wireless programmers or Internet-connected at-home monitors do not immediately protect the wireless links themselves and may not protect the integrity of communications. Controlled distribution of programmers cannot prevent insider access, and the availability of software radios can make the possession of a commercial wireless reprogrammer unnecessary. In essence, device manufacturers should not view external devices, like commercial programmers, as part of the trusted computing base for IMDs. Additionally, the trend toward increasing nominal read range allows flexible home monitoring for better safety, but it also increases the exposure of devices to attacks from nearby adversaries. Finally, for economic and safety reasons, certain IMDs typically contain non-rechargeable batteries and require surgery for replacement. Conventional approaches for security and privacy may facilitate trivial denial of service attacks against these batteries.

**Future directions and open problems.** Critical systems that rely on computing devices are already designed with great care. When these systems include *wireless* computing devices, additional precautions are necessary to ensure that the computing devices appropriately balance safety with convenience and do not introduce unacceptable risks. Medical device design is one such situation. Our research into implantable cardioverter defibrillators has demonstrated failure modes that do not appear to be addressed by some present-day design strategies

and certification processes.

Our work therefore leaves open a number of research problems. While there are a few obvious minor next steps, our research calls for much broader and innovative action. In concert with Halperin *et al.* [10], this work illuminates the need for a principled and deeper investigation into prevention mechanisms, detection mechanisms, audit mechanisms, deterrents, and methods that enhance patient awareness and ensure consent. Moreover, a fundamental challenge will be to develop methods that appropriately balance security and privacy with traditional goals such as safety and effectiveness. Our work provides a foundation for these explorations, on top of which we hope to see much subsequent innovation. Such innovations will become more crucial as the technologies and capabilities of implantable medical devices continue to evolve.

## ACKNOWLEDGMENTS

We thank Rick Adrion, Tom Anderson, Mark Corner, Matthew Garber, Robert W. Jackson, Barry Karas, Jim Kurose, Ed Lazowska, Mark McGovern, Jerry Saltzer, Stefan Savage, Prashant Shenoy, Joshua Smith, Bill Stasny, Paul van Oorschot, David Wagner, Matt Welsh, David Wetherall, Dan Yeager, Serge Zhilyaev, and the anonymous reviewers for their collective assistance, feedback, and generous support of this many-person, interdisciplinary project. This work was supported in part by National Science Foundation (NSF) grants CNS-0435065, CNS-0520729, CNS-0627529, and the NSF Research Experiences for Undergraduates (REU) program. Dr. Maisel is a U.S. Food and Drug Administration consultant. The opinions expressed in this paper are those of the authors and do not necessarily reflect the opinions, positions, practices, or policies of the FDA.

## REFERENCES

[1] A. Bellissimo, J. Burgess, and K. Fu. Secure software updates: disappointments and new challenges. In *Proceedings of USENIX Hot Topics in Security (HotSec)*, July 2006.

[2] H.-J. Chae, D. J. Yeager, J. R. Smith, and K. Fu. Maximalist cryptography and computation on the WISP UHF RFID tag. In *Proceedings of the Conference on RFID Security*, July 2007.

[3] S. Cherukuri, K. Venkatasubramanian, and S. Gupta. BioSec: A biometric based approach for securing communication in wireless networks of biosensors implanted in the human body. In *ICPP Workshops*, 2003.

[4] T. Drew and M. Gini. Implantable medical devices as agents and part of multiagent systems. In *Fifth International Joint Conference on Autonomous Agents and Multiagent Systems*, May 2006.

[5] Ettus Research, LLC. The universal software radio peripheral. http://www.ettus.com.

[6] K. Fu, S. Kamara, and T. Kohno. Key regression: Enabling efficient key distribution for secure distributed storage. In *Proceedings of the Symposium on Network and Distributed Systems Security*, February 2006.

[7] M. Fukomoto, M. Shinagawa, and T. Sugimura. A broad-band intrabody communication system with electro-optic probe. In *Proceedings of the First International Conference on Appliance Design 2003*, 2003.

[8] M. T. Goodrich, M. Sirivianos, J. Solis, G. Tsudik, and E. Uzun. Loud and clear: Human-verifiable authentication based on audio. Cryptology ePrint Archive, Report 2005/428, 2005. http://eprint.iacr.org/.

[9] N. M. Haller. The S/KEY one-time password system. In *Proceedings of the Symposium on Network and Distributed Systems Security*, February 1994.

[10] D. Halperin, T. S. Heydt-Benjamin, K. Fu, T. Kohno, and W. H. Maisel. Security and privacy for implantable medical devices. *IEEE Pervasive Computing, Special Issue on Implantable Electronics*, January 2008.

[11] T. S. Heydt-Benjamin, D. V. Bailey, K. Fu, A. Juels, and T. O'Hare. Vulnerabilities in first-generation RFID-enabled credit cards. In *Proceedings of Financial Cryptography and Data Security, 11th International Conference, LNCS 4886*, February 2007.

[12] D. E. Holcomb, W. P. Burleson, and K. Fu. Initial SRAM state as a fingerprint and source of true random numbers for RFID tags. In *Proceedings of the Conference on RFID Security*, July 2007.

[13] C. Karlof, N. Sastry, and D. Wagner. TinySec: A link layer security architecture for wireless sensor networks. In *Second ACM Conference on Embedded Networked Sensor Systems (SenSys 2004)*, November 2004.

[14] M. Kuhn. *Compromising emanations: eavesdropping risks of computer displays*. PhD thesis, Computer Laboratory, University of Cambridge, Wolfson College, 2003.

[15] J. Lackey and D. Hulton. The A5 cracking project: Practical attacks on GSM using GNU radio and FPGAs. In *Chaos Communication Camp*, 2007.

[16] N. G. Leveson and C. S. Turner. An investigation of the Therac-25 accidents. *Computer*, 26(7):18–41, 1993.

[17] Maine State Planning Office. Noise technical assistance bulletin #4. http://www.maine.gov/spo/landuse/docs/NoiseTABulletin.pdf, May 2000. Last viewed November 9, 2007.

[18] W. H. Maisel. Safety issues involving medical devices. *Journal of the American Medical Association*, 294(8):955–958, August 2005.

[19] W. H. Maisel, M. Moynahan, B. D. Zuckerman, T. P. Gross, O. H. Tovar, D.-B. Tillman, and D. B. Schultz. Pacemaker and ICD generator malfunctions: Analysis of Food and Drug Administration annual reports. *Journal of the American Medical Association*, 295(16):1901–1906, April 2006.

[20] J. M. McCune, A. Perrig, and M. K. Reiter. Seeing-is-believing: Using camera phones for human-verifiable authentication. In *IEEE Symposium on Security and Privacy*, pages 110–124. IEEE Computer Society, 2005.

[21] Medtronic. Maximo DR. http://www.medtronic.com/crm/performance/icd/7278-maximo-dr.html. Last viewed November 8, 2007.

[22] M. Meingast, T. Roosta, and S. Sastry. Security and privacy issues with health care information technology. In *Proceedings of the 8th Annual International Conference of the IEEE Engineering in Medicine and Biology*, pages 5453–5458, August 2006.

[23] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J. D. Tygar. SPINS: Security protocols for sensor networks. *Wireless Networks*, 8(5):521–534, Sept. 2002.

[24] R. L. Rivest. The RC5 encryption algorithm. In B. Preneel, editor, *Fast Software Encryption*, pages 86–96. Springer, 1995. (Proceedings Second International Workshop, Dec. 1994, Leuven, Belgium).

[25] A. P. Sample, D. J. Yeager, P. S. Powledge, and J. R. Smith. Design of a passively-powered, programmable platform for UHF RFID systems. In *IEEE International Conference on RFID 2007*, March 2007.

[26] H. Savci, A. Sula, Z. Wang, N. S. Dogan, and E. Arvas. MICS transceivers: regulatory standards and applications [medical implant communications service]. In *Proceedings of IEEE SoutheastCon 2005*, pages 179–182, April 2005.

[27] J. R. Smith, A. P. Sample, P. S. Powledge, S. Roy, and A. Mamishev. A wirelessly-powered platform for sensing and computation. In *8th International Conference on Ubiquitous Computing (Ubicomp 2006)*, pages 495–506, Orange Country, CA, USA, September 2006.

[28] D. Spill and A. Bittau. BlueSniff: Eve meets Alice and Bluetooth. In *Proceedings of USENIX Workshop on Offensive Technologies (WOOT)*, August 2007.

[29] F. Stajano and R. J. Anderson. The resurrecting duckling: Security issues for ad-hoc wireless networks. In *Proceedings of Security Protocols, 7th International Workshop, LNCS 1796*, pages 172–194, 1999.

[30] The Free Software Foundation. The GNU software radio. http://www.gnu.org/software/gnuradio.

[31] K. K. Venkatasubramanian and S. K. S. Gupta. Security for pervasive health monitoring sensor applications. In *Proceedings of 4th International Conference on Intelligent Sensing and Information Processing (ICISIP)*, pages 197–202, December 2006.

[32] S. Warren, J. Lebak, J. Yao, J. Creekmore, A. Milenkovic, and E. Jovanov. Interoperability and security in wireless body area network infrastructures. In *Proceedings of the 2005 IEEE Engineering in Medicine and Biology 27th Annual Conference*, 2005.

[33] L. Zhong, D. El-Daye, B. Kaufman, N. Tobaoda, T. Mohamed, and M. Liebschner. OsteoConduct: Wireless body-area communication based on bone conduction. In *Proc. Int. Conf. Body Area Networks (BodyNets)*, June 2007.