Short Paper: Speaking the Local Dialect: Exploiting differences between IEEE 802.15.4 Receivers with Commodity Radios for fingerprinting, targeted attacks, and WIDS evasion

Ira Ray Jenkins
jenkins@cs.dartmouth.edu
Dept. of Computer Science
Dartmouth College
Hanover, New Hampshire
USA

Rebecca Shapiro
bx@cs.dartmouth.edu
Dept. of Computer Science
Dartmouth College
Hanover, New Hampshire
USA

Sergey Bratus
sergey@cs.dartmouth.edu
Dept. of Computer Science
Dartmouth College
Hanover, New Hampshire
USA

Travis Goodspeed
travis@radiantmachines.com
Straw Hat

Ryan Speers
ryan@riverloopsecurity.com
River Loop Security, LLC

David Dowd
david@riverloopsecurity.com
River Loop Security, LLC

ABSTRACT

Producing IEEE 802.15.4 PHY-frames reliably accepted by some digital radio receivers, but rejected by others—depending on the receiver chip's make and model—has strong implications for wireless security. Attackers could target specific receivers by crafting "shaped charges," attack frames that appear valid to the intended target and are ignored by all other recipients. By transmitting in the unique, slightly non-compliant "dialect" of the intended receivers, attackers would be able to create entire communication streams invisible to others, including wireless intrusion detection and prevention systems (WIDS/WIPS).

These scenarios are no longer theoretic. We present methods of producing such IEEE 802.15.4 frames with *commodity digital radio chips* widely used in building inexpensive 802.15.4-conformant devices. Typically, PHY-layer fingerprinting requires software-defined radios that cost orders of magnitude more than the chips they fingerprint; however, our methods do not require a software-defined radio and use the same inexpensive chips.

Knowledge of such differences, and the ability to fingerprint them is crucial for defenders. We investigate new methods of fingerprinting IEEE 802.15.4 devices by exploring techniques to differentiate between multiple 802.15.4-conformant radio-hardware manufacturers and firmware distributions. Further, we point out the implications of these results for WIDS, both with respect to WIDS evasion techniques and countering such evasion.

WiSec'14, July 23–25, 2014, Oxford, United Kingdom

http://dx.doi.org/10.1145/2627393.2627408.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*

Keywords

IEEE 802.15.4; ZigBee; wireless sensor networks; security

1. INTRODUCTION

Wireless sensor networks (WSN) represent a massive and rapidly growing technology sector. Market research estimates 1 billion radio frequency integrated circuit (RFIC) devices will be deployed by 2017 [17], the majority of which will be IEEE 802.15.4 [2] and ZigBee [29] standards compliant.

Due to the increasing dependence on these digital radios it is important to understand their stacks from the ground up. In particular, manipulations of PHY- and LNK-frames achievable with commodity means—and other commodity stacks' reactions to them—are of special interest. Overlooked capabilities of commodity hardware leads to nasty surprises for the defenders of a deployed base previously considered reasonably secure. For example, discovery of methods to inject arbitrary crafted 802.11 LNK-frames with commodity 802.11 hardware around 2005 led to the embarrassing "Month of Kernel Bugs" (MoKB) in 2006 that exposed multiple vulnerabilities in Wi-Fi drivers across all operating systems and in many embedded implementations.

We demonstrate that commodity 802.15.4 digital radios are capable of producing PHY-frames that differ in appearance between various 802.15.4 receivers; while being accepted as valid by some receivers, these frames may be rejected by others—depending on the radio chip's make (and, occasionally, firmware). Using this physical-frame-level technique the attacker can craft and broadcast "shaped charges" to covertly communicate with target nodes by constructing a "dialect" of IEEE 802.15.4 PHY-frames intelligible to only

these nodes. This allows attackers to bypass WIDS/WIPS systems that utilize different digital radio receivers, as a monitor(s), than that of the network nodes they protect.

The purpose of this work is to expand the state-of-the-art in IEEE 802.15.4 physical-layer manipulation achievable with commodity 802.15.4/ZigBee devices, enabling device identification, targeted attacks, and WIDS/WIPS bypasses. We have built an experimental framework, code-named Isotope, around commodity hardware and open-source software. We have also developed several techniques effective in differentiating between multiple devices' hardware.

The remainder of this paper is organized as follows: Section 2 discusses previous work and provides context for our contributions; Section 3 provides a brief primer on the IEEE 802.15.4 standard and introduces the frame crafting techniques we have developed; Section 4 describes our experimental setup; Section 5 reveals our results; and Section 6 offers concluding remarks and a nod toward future work.

2. PREVIOUS WORK

Our work extends previous work on *active fingerprinting* from our lab [6, 10, 12, 26]. It also harkens back to the classic work on evading intrusion detection and prevention systems (IDS/IPS) [16, 23] that exploited differences in network streams reassembly by the attack targets and the IDS/IPS protecting them—which have since been generalized as *parser differential* attacks [19, 25].

Our work is similar to independently created results by Ramsey, Mullins, and Kulesza [20, 24]. They showed shortening the preambles made the packets non-receivable to some sniffers. We present such a result, in addition to other observations at the 802.15.4 PHY-layer, but using only commodity hardware and open-source software.

2.1 Digital Radio Fingerprinting

Fingerprinting endeavors to exploit unique characteristics in the digital circuitry or firmware implementation of a device. Slight imperfections in the radio circuitry, introduced during the manufacturing process, might be detectable during radio transmissions. In addition, bugs or deviations from the standard in the firmware implementation may also be observable during radio operation and can act as a fingerprint. For a more detailed understanding, Danev, Zanetti, and Capkun provide a thorough survey of the state-of-the-art in wireless fingerprinting [11].

In passive fingerprinting methods, a third party attempts to unobtrusively sniff the communications channel [13, 14, 18]. Unique signals or transmission timing may be considered a fingerprint. Naturally, this approach is often lossy or error prone due to the potential lack of traffic over the wire or interference from the multiple layers of the radio stack [23]. Alternatively, active techniques attempt to interact with a device, often by sending specially crafted requests, in hopes of eliciting a response [6, 10]. Both the data contained in the response and the response itself can be considered a fingerprint.

Applications of Fingerprinting. Attackers have used fingerprinting techniques to find systems known to be vulnerable, see through defensive deceptions such as false bannering or redirecting honeypots [5], and to impersonate trusted nodes on a network. Not surprisingly, as soon as fingerprinting techniques became a part of standard TCP/IP network reconnaissance an arms race ensued with tools offering functionality to deceive fingerprinting techniques by imitating known signatures. Meanwhile, defenders use fingerprinting techniques to identify nodes on their network, both benign and malignant, and to find vulnerable software, firmware, and hardware combinations. The IEEE 802.15.4 and ZigBee standards offer no exception to this rule. By design, these are commodity technologies (in particular, much more so at their origins than 802.11/Wi-Fi).

Symbols: 8	2	2		variable
Preamble	SFD	Frame length (7 bits)	Reserved	PSDU
SHR		PHR		PHY payload

Figure 1: An IEEE 802.15.4 standard physical frame.

3. METHODS

In this section, we look at the IEEE 802.15.4 standard and describe the receiver fingerprinting and targeting techniques we have developed.

3.1 IEEE 802.15.4 Standard

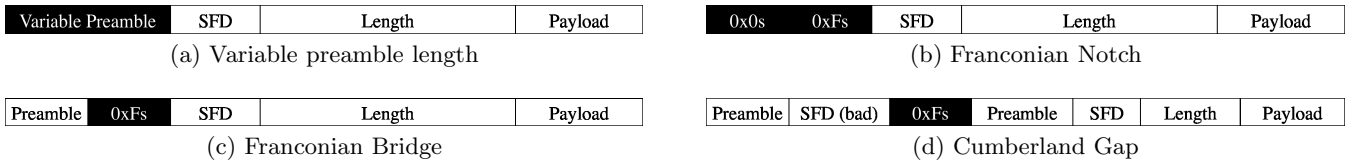
The IEEE created the 802.15 workgroup for Wireless Personal Area Networks (WPAN) in the early 2000s to establish standards for Layers 1 and 2 (physical and link, respectively). The IEEE 802.15 workgroup defined standards that include 802.15.1, a derivative of Bluetooth intended for general WPANs, and 802.15.4, designed for low-rate WPANs (LR-WPANs). LR-WPANs are attractive for low-power, low-range, low-bandwidth, and low-cost applications of wireless networking, particularly for industrial control and embedded systems.

ZigBee is a Layer 3 (network layer) specification which layers on top of 802.15.4 and is more well-known. While ZigBee is ripe for investigation in many different forms of fingerprinting; this paper focuses on the layer beneath ZigBee—the IEEE 802.15.4 standard.

In the IEEE 802.15.4 standard, the smallest amount of information that can be sent over the air is four bits, or a symbol. The standard defines four types of physical frames: beacon, data, acknowledgement, and command. The standard physical frame layout, for all frames, is shown in Figure 1. A standard frame consists of a synchronization header (SHR), a physical layer (PHY) header (PHR), and a payload within the physical service data unit (PSDU). The physical frames differ in their payload, but all contain a standard SHR and PHR. The SHR comprises an 8-symbol preamble of zeros (0x0) and the start-of-frame delimiter (SFD), which must be 0xA7. This header, as its name implies, serves to synchronize the receiving radio with the transmitting radio so that symbols are correctly pulled out of the signal. The frame length, a 7-bit number representing the number of octets in the physical payload, and a single reserved bit compose the PHR. The payload follows the length and contains all the data for Layer 2 and higher. Each type of physical frame requires a different payload structure.

3.2 Crafting Physical Frame Headers

Before introducing the designed methods, it should be noted that many commodity radios cannot craft arbitrary physical frame headers, SHR and PHR. By design, the ra-



(a) Variable preamble length

(b) Franconian Notch

(c) Franconian Bridge

(d) Cumberland Gap

Figure 2: New fingerprinting frames.

dio hardware manages the frame headers to assure proper functionality. In order to fully control a physical frame's contents, we make use of our good neighbor Travis Goodspeed, et al's packets-in-packets (PIP) frame-injection technique [15].

The 802.15.4 standard requires the SFD to be 0xA7. If an 802.15.4-compliant radio receives an SFD of any other value, the receiving radio resets itself into a fresh receiving state, listening again for a new SHR; however, some radios permit us to specify the SFD value via a register, which allows us to transmit frames with non-compliant SFDs. Any receivers expecting the standard SFD will reset themselves after seeing the unexpected symbols. The transmitting radio, however, will continue to send the remainder of the frame. If the remainder of the frame contains a standard SHR the receiver will think it is receiving a fresh packet. In this way, we are able to transmit a non-standard physical frame that contains a fully-standard physical frame, or a packet in a packet.

3.3 Fingerprinting Techniques

Here we will describe four new techniques for fingerprinting IEEE 802.15.4 stacks, with a focus on the physical layer. Each technique is active—a stimulus frame with a non-standard physical-layer header is transmitted and the target's response or lack thereof is recorded. Our hypothesis is that we can distinguish different radio chipsets by which type of stimulus packets they are willing to receive. To determine whether a given chipset has indeed received a packet, we send a frame whose payload triggers a response by a higher layer—such as beacon request. If we receive the correct response to our stimulus, we assume that our crafted frame was received.

A Variable Preamble Length. While the IEEE 802.15.4 standard defines the preamble length to be eight zero (0x0) symbols some radios might accept frames with fewer than the stated number, while others do not. Figure 2a shows the general layout of a frame generated to test a target's response to non-standard preambles. The aim of this technique is to measure the number of zero symbols, before the SFD, a chipset requires in order to accept a frame.

A Franconian Notch. According to the IEEE 802.15.4 specification, a preamble field should contain eight zero (0x0) symbols. However, some chipsets may accept non-standard preambles. For example, the CC2420 [28] can be programmed to ignore some of the least significant symbols in the synchronization header to help it be more resilient to noise. Figure 2b shows the physical frame crafted for the Franconian Notch[1] method. Here we modulate each subsequent symbol of the standard preamble from 0x0 to 0xF , going from all zeros (0x0s) to six 0xF symbols (thus introducing a "notch"). The aim of this technique is to measure the number of invalid preamble symbols a radio is willing to accept.

A Franconian Bridge. Inspired by the previous approaches, the Franconian Bridge method "spans the gap" between the variable preamble length and Franconian Notch techniques. As shown in Figure 2c, the Franconian Bridge investigates how a target responds to having a varying number of 0xF symbols placed between the fully-standard preamble and the SFD. Technically, this will evaluate a radio's behavior in the presence of a seemingly non-standard SFD.

A Cumberland Gap. The Cumberland Gap[2] technique, as seen in Figure 2d, measures how a target behaves with respect to receiving a standard frame immediately after receiving a valid preamble and an invalid SFD, thus introducing a "gap" between a bad frame and a good frame.

It is important to remember that when radios are listening for data, they read whatever they find into a symbol. Therefore, it is quite common for a radio to be prepared to accept a frame when it is merely listening to interference and reading garbage as symbols. There are a few discrete states that a radio state machine has to go through when finding an SFD. In this method, we intentionally make the SFD very close to the standard to nudge the receiver as close as possible to the state in which it receives a full frame without outright telling it to take the remainder of the frame. When the incorrect SFD arrives, the chip goes back to listening for a preamble—we seek to measure the timing of this behavior. The fewer symbols that we can inject and still get a response may imply a faster turn-over time, and might also signify a fingerprint.

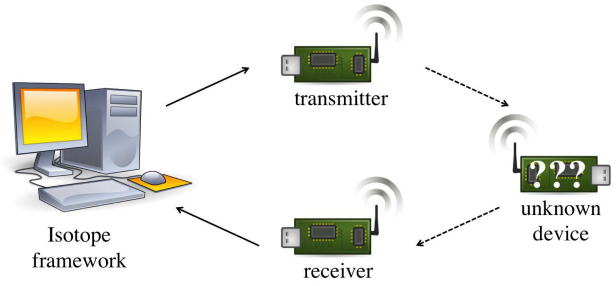


Figure 3: The fingerprinting testbed.

[1]The Franconian Notch is a mountain pass through the White Mountains of New Hampshire.

[2]The Cumberland Gap is a mountain pass through the Appalachian Mountains between Tennessee, Kentucky, and Virginia.

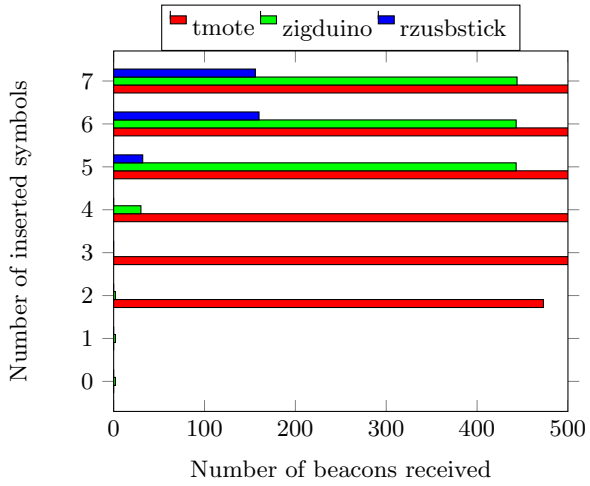


Figure 4: Variable preamble results.

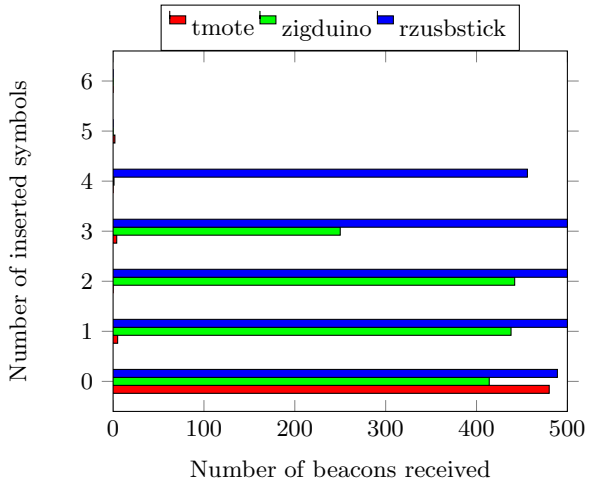


Figure 5: Franconian Notch results.

4. EXPERIMENTAL SETUP

To test the functionality of our proposed fingerprinting methods, we built a testbed to examine how different IEEE 802.15.4 stacks respond to the types of non-standard physical headers previously described.

4.1 Testbed Layout

Our testbed consists of only commodity hardware and open-source software. As shown in Figure 3, two IEEE 802.15.4-conformant radios are connected (via serial over USB) to a single workstation running Isotope, our fingerprinting software. Isotope is a Python framework that utilizes the open source libraries Scapy [4], to build 802.15.4 physical frames, and KillerBee [3], to configure the radios, monitor communications traffic, and inject arbitrary frames. One radio is used solely to transmit crafted frames and the other radio is used to sniff all traffic on a particular channel. The third, unknown, device is setup to listen on a specific channel and respond to beacon requests.

4.2 Hardware and Software

We experimented with several different radio devices, namely Zigduinos [21], RZUSBsticks [9], and the popular (but now discontinued) Tmote Sky [22]. Each of these devices contain different on-board radio chips, namely an Atmel ATmega128RFA1 [8], an Atmel AT86RF230 [7], and a Chipcon CC2420 [28], respectively. Each device was tested with the GoodFET [1] open-source firmware.

5. RESULTS

A variable preamble length. Figure 4 shows the results of varying the number of preamble symbols. Clearly, the Tmote device responds to the fewest number of preamble symbols. It is possible that this is by design. Remember, the Tmote contains a CC2420 radio chip which allows a programmable number of preamble bits to be accepted. Assuming normal function, it seems obvious that the Tmote is distinguishable from the Zigduino and RZUSBstick.

A Franconian Notch. Figure 5 showcases the results of transforming the preamble from 8 zero (0x0) symbols to 6 0xF symbols. Zero (0) on the Y-axis represents a fully standard physical frame, with zero 0xF symbols present. It appears as though the Tmote, previously loose with the standard, is now fully compliant. Since the Tmote previously accepted fewer preamble symbols, this could be an artifact of the radio interpreting the additional 0xF symbols as an invalid SFD, or it could have to do with the RF demodulator's sync circuit being thrown out of state by the additional bit transitions. At the other end of the spectrum, we find the RZUSBstick also stands out by accepting as many as 4 0xF symbols within the preamble. Both the Tmote and RZUSBstick look to be distinguishable from each other and the Zigduino.

A Franconian Bridge. The results for the Franconian Bridge method are shown in Figure 6. Recall that this technique inserts garbage between a valid preamble and a valid SFD. Ideally, a radio would interpret the garbage as an invalid SFD. As in the previous method's results, the Tmote strictly adheres to the standard; while, the RZUSBstick accepts up to 5 garbage symbols interposed between the preamble and SFD. We also find the Zigduino occasionally responding to as many as 6 symbols.

A Cumberland Gap. The results for the Cumberland Gap method, seen in Figure 7, do not seem encouraging. It appears as though the Tmote has the fastest turnaround time, responding to any number of garbage symbols spliced between a bad SFD and a good packet. Meanwhile, the RZUSBstick maintains the slowest. This may be attributed to fact that the RZ uses an on-board antenna versus the Tmote and Zigduino's external antennas.

6. CONCLUSIONS

With the number of wireless sensor networks exploding, a large portion being IEEE 802.15.4 and ZigBee devices, it is essential that we be able to secure and protect these devices and networks for mission-critical systems. Fingerprinting these radio devices is a first step along the path to achieving that security. Device identification, both passive and active, has been used on many other wireless network protocols.

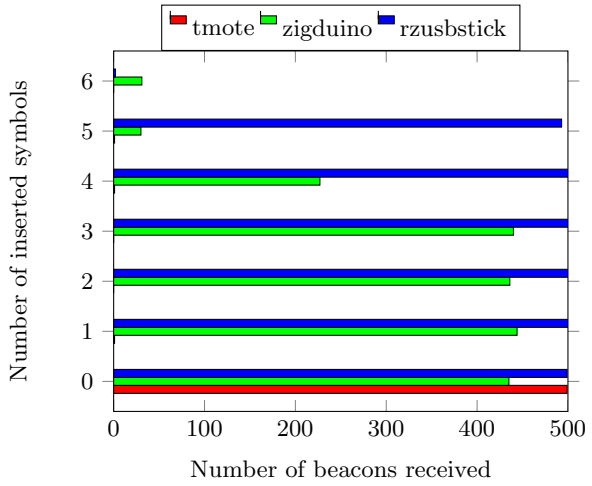


Figure 6: Franconian Bridge results.

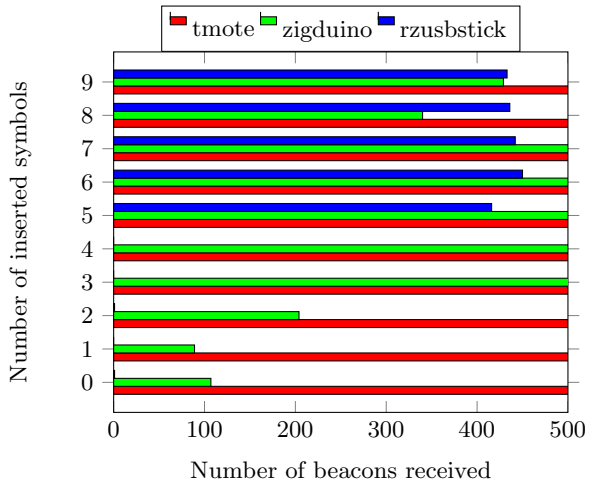


Figure 7: Cumberland Gap results.

Our work seeks to apply it to IEEE 802.15.4-conformant radio devices. By accurately identifying different devices, we have another tool, on-top of PKI authentication schemes, for verifying trusted nodes in a network. Similarly, by analyzing how frames and packets make their way through the firmware and radio circuitry, it is possible that we may uncover hidden vulnerabilities and attack vectors.

With preliminary results, it appears that the Tmote devices, with the Chipcon CC2420 radio chipset, and the RZUSBsticks, with the Atmel AT86RF230 radio chipset, are differentiable (at least between themselves and a Zigduino receiver). The Tmotes clearly respond to very non-standard preamble lengths, whether by design or flaw; however, the same devices seem to be very strict on preamble and SFD content. It appears that the RZUSBsticks accept very non-standard preamble and SFD content. Currently, the CC2420 chips look like the top contender so far to avoid WIDS detection.

6.1 Future Work

We feel this work is ripe for research. There are many more possible firmware and hardware combinations to test—we have really only just begun. Moving forward, it is critical to obtain and test additional devices (both firmware and radio chips). The Tmote devices, our main contender for WIDS evasion, are no longer in production. We have recently begun producing a new device, the ApiMote, based on the CC2420. We suspect the two devices will react similarly; however, those results are forthcoming. It will also be necessary to conduct additional tests in high noise environments and within an isolated RF chamber. Of course, our software framework, Isotope, will also require some additional refinements to make it more robust. Typically, in device identification, a database of fingerprints is used in combination with some sort of machine learning method to analyze and evaluate fingerprint matches. Our current work constitutes only the first stage of identifying possible fingerprints. We anticipate potential vulnerabilites lie within these device-specific dialects, such as possible length overflows. Lastly, we would like to explore the potential for WIDS evasion by these commodity radios [27].

7. ACKNOWLEDGEMENTS

This research was supported in part by the National Science Foundation, under Grant Award Number 1016782, and the Department of Energy, under Grant Award Number DE-OE0000097. This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

8. REFERENCES

[1] GoodFET. `http://goodfet.sourceforge.net`.
[2] IEEE Computer Society. `http://www.ieee.org`.
[3] KillerBee. `http://code.google.com/p/killerbee/`.
[4] Scapy. `http://www.secdev.org/projects/scapy/`.
[5] Tiny Honeypot. `http://freecode.com/projects/thp`.
[6] C. Arackaparambil, S. Bratus, A. Shubina, and D. Kotz. On the reliability of wireless fingerprinting using clock skews. In *Proceedings of the 3rd ACM conference on Wireless network security*, pages 169–174. ACM, 2010.
[7] Atmel Corporation. AT86RF230 datasheet. `http://www.atmel.com/Images/doc5131.pdf`.
[8] Atmel Corporation. ATmega128RFA1 datasheet. `http://www.atmel.com/Images/doc8266.pdf`.
[9] Atmel Corporation. RZUSBstick. `http://www.atmel.com/tools/RZUSBSTICK.aspx`.
[10] S. Bratus, C. Cornelius, D. Kotz, and D. Peebles. Active behavioral fingerprinting of wireless devices. In *Proceedings of the first ACM conference on Wireless network security*, pages 56–61. ACM, 2008.

[11] B. Danev, D. Zanetti, and S. Capkun. On physical-layer identification of wireless devices. *ACM Computing Surveys (CSUR)*, 45(1):6, 2012.

[12] D. D. Dowd. *Isotope: Active Behavioral Fingerprinting of IEEE 802.15.4 Devices*. Senior honors thesis, Dartmouth College, Computer Science, August 2012.

[13] J. P. Ellch. Fingerprinting 802.11 devices. Master's thesis, Naval Postgraduate School, 2006.

[14] J. Franklin, D. McCoy, P. Tabriz, V. Neagoe, J. V. Randwyk, and D. Sicker. Passive data link layer 802.11 wireless device driver fingerprinting. In *Proceedings of the 15th USENIX Security Symposium*, pages 167–178, 2006.

[15] T. Goodspeed, S. Bratus, R. Melgares, R. Shapiro, and R. Speers. Packets in Packets: Orson Welles' In-Band Signaling Attacks for Modern Radios. In *5th USENIX Workshop on Offensive Technologies (WOOT)*, pages 54–61, August 2011.

[16] M. Handley, V. Paxson, and C. Kreibich. Network Intrusion Detection: Evasion, Traffic Normalization, and End-to-End Protocol Semantics. In *Proceedings of the USENIX Security Conference*, 2001.

[17] M. Hatler, D. Gurganious, and C. Chi. 802.15.4 & ZigBee: Expanding markets, growing threats. Technical report, A Market Dynamics report (9th edition), 2012.

[18] S. Jana and S. K. Kasera. On fast and accurate detection of unauthorized wireless access points using clock skews. *Mobile Computing, IEEE Transactions on*, 9(3):449–462, 2010.

[19] D. Kaminsky, L. Sassaman, and M. Patterson. PKI Layer Cake: New Collision Attacks Against The Global X.509 CA Infrastructure. Black Hat USA, August 2009.

[20] N. Kulesza, B. W. P. Ramsey, and B. E. Mullins. Wireless intrusion detection through preamble manipulation. In *Proceedings of the 9th Int'l Conf on Cyber Warfare and Security*, pages 132–139, 2014.

[21] Logos Electromechanical LLC. Zigduino. `http://logos-electro.com/zigduino/`.

[22] Moteiv Corporation. Tmote Sky datasheet. `http://www.snm.ethz.ch/snmwiki/pub/uploads/Projects/tmote_sky_datasheet.pdf`.

[23] T. H. Ptacek and T. N. Newsham. Insertion, evasion, and denial of service: Eluding network intrusion detection. Technical report, Secure Networks Inc, 1998.

[24] B. W. P. Ramsey and B. E. Mullins. In J. Butts and S. Shenoi, editors, *Critical Infrastructure Protection*, IFIP Advances in Information and Communication Technology, pages 63–79. Springer.

[25] L. Sassaman, M. L. Patterson, S. Bratus, and M. E. Locasto. Security Applications of Formal Language Theory. *IEEE Systems Journal*, 7(3), September 2013.

[26] R. Speers. IEEE 802.15.4 Wireless Security: Self-Assessment Frameworks. Technical Report TR2011-687, Dartmouth College, Computer Science, Hanover, NH, June 2011.

[27] R. Speers, J. Vazquez, and S. Bratus. Making (and Breaking) an IEEE 802.15.4 WIDS. In *Troopers*, 2014.

[28] Texas Instruments. CC2420 datasheet. `http://www.ti.com/lit/ds/symlink/cc2420.pdf`.

[29] Zigbee Alliance. ZigBee. `http://www.zigbee.org`.