A Toolbox for RFID Protocol Analysis

Roel Verdult *Gerhard de Koning Gans* *Flavio D. Garcia*

Institute for Computing and Information Sciences.
Radboud University Nijmegen, The Netherlands.
{rverdult,gkoningg,flaviog}@cs.ru.nl

Abstract **- Many RFID tags and contactless smart cards use proprietary security mechanisms for authentication and confidentiality. There are several examples in the literature showing that once these mechanisms have been reverse engineered, their security turns out to be unsatisfactory. Since the use of these tags is quickly expanding to access control and ticketing systems, it is important to independently assess their security. In this paper, we propose three tools for the analysis of RFID protocols. These tools facilitate message eavesdropping and emulation of both tags and readers. The tools focus on high frequency tags but one of them also supports low frequency. These tools are fully programable and allow for quick prototyping, testing and debugging of new RFID protocols. All the software, firmware and hardware we have developed that is described here is open source and open design.**

I. Introduction

Radio Frequency Identification (RFID) is one of the most pervasive technologies nowadays. It was first introduced for identification purposes only, but quickly expanded to other applications like transport ticketing systems and access control. The security and privacy of these systems is often overlooked. This is, to a good extend, due to the fact that it is hard to know what are the underlying security mechanisms employed. A surprisingly large number of access control systems use the tag's unique identifier (UID) as their only security mechanism. Moreover, large scale ticketing systems use simple memory cards [1–4] for fare collection. This type of cards lack any cryptographic capabilities and therefore the security of these systems relies on UID blacklisting mechanisms. When portable tag-emulating devices are available [5–7], these security mechanisms become obsolete. Manufacturers often claim that their tags provide 'state-of-the-art', 'field-proven' or 'unbreakable' security, but it is hard to know what this means and how much security you actually get. The widely used RFID communication standards like [8–12] define the low-level transmission layers. However, these standards do not include any details of the secure communication layer. Semiconductor companies are inclined to create ad-hoc RFID designs that use proprietary protocols and cryptographic algorithms [13–21]. Such designs are often kept secret to provide security-through-obscurity. It has been shown many times that without feedback from the scientific community, it is hard to build secure algorithms [22, 23]. There are numerous examples in the literature [24–39] showing that once the secrecy of an algorithm is lost, so is its security. As long as RFID tags do not comply with open and community-reviewed encryption standards, the security of these tags need to be independently assessed. In order to perform these assessments, we need tools to analyze the underlying security protocols. While designing new RFID products and protocols, it is also useful to have a set of tools at hand for easy protocol prototyping, testing and debugging.

1.1 Our contribution

We have developed a toolbox for the analysis of RFID protocols. The toolbox consists of three tools: the Ghost; the Proxmark; and the Libnfc. Each of these tools have different characteristics and capabilities. We focus on high frequency tags, although the Proxmark is also capable of modulating low frequency.

The Ghost is a portable and inexpensive tag emulator. It was entirely developed at our university. This means that we have developed the hardware and firmware of the device. The Ghost was used in [33] to reverse engineer the Crypto1 cipher and the authentication protocol of the Mifare Classic. The Proxmark is slightly more expensive (€200) but much more powerful than the Ghost. It is fully programmable and can operate both as tag or as reader. The Proxmark is so versatile that it can even be used for quick prototyping of non-standard protocols like reader-to-reader anti-collision [40], the fair anti-collision protocol [41] and other protocols in the literature [42, 43].

For the Proxmark, we have developed the firmware that does the Modified-Miller and Manchester coding which allows it to operate in ISO/IEC 14443-A compatible mode. The Proxmark was used to implement the attacks on the Mifare Classic proposed in [33, 44–48]. Next, we optimized the eavesdropping capabilities of the Proxmark to intercept ISO/IEC 14443-B communication. This was used to reverse engineer and attack the proprietary stream cipher of CryptoRF [36, 49, 50]. Furthermore, we added support for Amplitude Shift Keying (ASK) modulation and the 1 out of 4 data coding mode as specified in the ISO/IEC 15963. This functionality was used to perform a security analysis of the widely used iClass cards [39, 51]. Recently, we added support for the Hitag2 vehicle immobilizer transponders which operate on low frequency at 125 kHz. The practical experiments of [52] show how use this feature to recover the secret key from a car key within minutes.

Libnfc is a software package that works with most commercially-available NFC readers. Libnfc is capable of implementing most of the aforementioned attacks using only off-the-shelf hardware. For all three tools we have developed software for message eavesdropping, injection and analysis. All this software (and hardware) is open-source (design) and publicly available[1] under GNU General Public Licence. Furthermore, the Libnfc library was used to reveal several vulnerabilities in modern NFC cell-phones [53]. The experiments show that an adversary can spread malware with a low-cost reader based on Libnfc.

1.2 Related work

The OpenPCD tool is part of an open-source and open-design high frequency emulator. The first tools of this project consisted of the OpenPCD and the OpenPICC which covered reader and tag emulation, respectively. Currently, there is a second version which replaces both tools, the OpenPCD 2. This tool covers both reader and tag emulation. Also, this new version supports sending arbitrary bits such that attacks like the ones described in [47, 48] can be mounted. All OpenPCD devices highly depend on commercial RFID baseband chips. This is not useful when research demands access beyond standardized modulations, encodings and anti-collision protocols. The OpenPCD 2 supports the Libnfc utilities described in Section 5.3.

The RFID guardian [54] is, in principle, capable of doing message eavesdropping and injection. Since it was designed for privacy protection, it is big and expensive ($>$ 1000 USD) and it might be overkill for the purpose of protocol analysis. To our knowledge this project is discontinued since 2009. Throughout this paper we focus on contact-

[1] http://www.cs.ru.nl/~flaviog/tools.html

less smart cards. For contact based smart cards there are comparable tools available in the literature [55, 56] which allows eavesdropping, emulation, man-in-the-middle attacks [57] and fast querying by using a dedicated FPGA.

II. Radio Frequency IDentification

There are many standards on RFID technology and a high variety of different frequencies are available for RFID applications. The most used frequencies are in the low frequency band at 125-134 kHz and in the high frequency band at 13.56 MHz. We focus on the three lowest layers of the OSI model, namely the *network layer*, *data link layer* and the *physical layer*. These layers facilitate the communication up to the bit level of messages that are sent over the air. The most relevant standards that fall within the scope of this paper are ISO/IEC 14443 and 15693 [10, 11].

An RFID reader creates an electro-magnetic field by generating a 13.56 MHz carrier wave. This carrier provides power to a card and is used for communication at the same time. The reader-to-card communication is in most cases achieved by interrupting the carrier wave for very short periods (a couple of milliseconds). A contactless card that is near the reader gets powered by induction.

At the physical layer of the communication, the sender has to transform the information to an analog signal. This analog signal is then captured at the receiver side and transformed back to the original digital message that was sent. The main steps of this transformation consist of encoding and modulation. It is important that these steps can be reversed by first demodulating and then decoding the signal. Sometimes the reader-to-card encoding is done differently than the card-to-reader encoding. For instance, in ISO 14443-A, Modified Miller encoding is used for reader-to-card communication and Manchester encoding is used for card-to-reader communication. This standard is widely used in many RFID systems, e.g. the Mifare Classic chip from NXP follows to a large extend this standard.

2.1 Modified Miller encoding

Modified Miller encoding is commonly used in the reader-to-card communication. As its name suggests, Modified Miller is a slightly adapted version of regular Miller encoding. In Miller encoding bits are encoded by making transitions between two states of communication. In case of ISO/IEC 14443-A, this means that the reader differentiates over two amplitude levels. The transitions between these levels, a high and low level, are used to encode data bits. A fixed elementary time unit (ETU) is used to indicate the bit length or bit period.

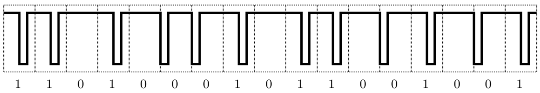

Figure 1 - Modified Miller encoding

ISO/IEC 14443-A uses 100% Amplitude Shift Keying (ASK) which means that the carrier drops out completely for some small period of time. Modified Miller is designed to make short drops in order to keep the encoded signal high as much as possible. This is needed to keep the card powered. A ‘0’ is encoded by a continuous high signal except for the case of consecutive zeros. Subsequent zeros are encoded by a drop right at the start of an ETU, see Figure 1. A ‘1’ is always encoded by a short drop halfway an ETU.

2.2 Manchester encoding

The card also needs to communicate back to the reader. A common encoding technique that is used for this is Manchester encoding, which is applied using On-Off Keying (OOK). 100% ASK comes down to a subcarrier that is switched on or off and therefore it is equivalent to OOK. This on-off keying of the subcarrier is also known as load modulation. A 847.5 kHz subcarrier is used in card-to-reader communication. The encoding itself is straightforward as a ’0’ is encoded by load modulation during the first half and a ‘1’ is encoded by load modulation during the second half of an ETU.

2.3 Modulation techniques

There are many modulation techniques and in this section we discuss the most common ones. Modulation is the technique of embedding a signal into a carrier wave. This signal can either be discrete or continuous. Modulation of a continuous signal (see Fig. 2), or so-called analog modulation, is for instance used in radio broadcasting where the analog audio signals are modulated using Frequency Modulation (FM) or Amplitude Modulation (AM). In RFID, discrete (or digital) modulation is used. The signal that is being modulated consists only of zeros and ones. For amplitude modulation, this results in a transmission signal that has only two amplitude levels (Fig. 3), unlike the continuous change of the amplitude level in analog modulation (Fig. 2). All these modulation techniques use the characteristics of a waveform. The frequency of a wave is the number of periods that occur within one second. One period corresponds to one cycle of the wave. The frequency of a wave is expressed in Hertz, e.g. 1 Hz = 1 cycle/second. The amplitude of a wave is the deviation from its average value. The bigger the amplitude, the more energy the wave carries. The phase of a wave is the initial angle at the origin of a sinusoidal function. Changing the phase of a wave can be seen as shifting the wave in time. A carrier wave c can be described by the sinusoidal function c $= A \cdot \sin(\omega t + \phi)$ where A is the amplitude, ω is the frequency and ϕ is the phase. These three parameters change the characteristics of the wave. Also, differentiations in an observed wave can be seen as changing A, ω and ϕ values. When the sender is able to introduce these changes, and the receiver is able to detect these changes, it is possible to communicate information. Altogether, we see three basic ways to influence the radio communication. First, changing the amplitude A, this is known as Amplitude Shift Keying (ASK). Then, changing the frequency of the signal, which is known as Frequency Shift Keying (FSK). And finally, changing the phase of the signal, this is known as Phase Shift Keying (PSK).

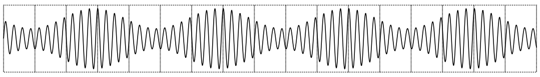

Figure 2 - Analog Amplitude Modulation

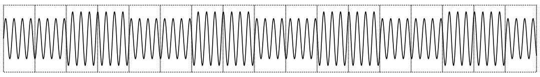

Figure 3 - Digital Amplitude Modulation

2.3.1 Amplitude Shift Keying

ASK uses changes in the amplitude to modulate a digital or analog signal into the carrier wave, see Figure 3. The most simple form of modulation, it basically comes down to switching the carrier wave on and off.

2.3.2 Frequency Shift Keying

In FSK, the frequency of the carrier wave changes over time in order to communicate information. In its most simple form it is called 2-FSK since two frequencies are used. There are several variants of FSK that use more frequencies which allows to communicate more than one bit of information during one ETU. The more complex 16-FSK uses 16 different frequencies where each frequency represents 4 bits of information. The effect of changing the frequency of the carrier wave and an example of 2-FSK is shown by Figure 4.

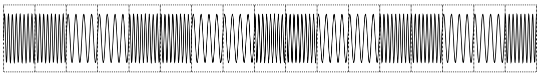

Figure 4 - Frequency Shift Keying

2.3.3 Phase Shift Keying

In PSK, the phase of the carrier wave changes over time. In line with ASK and FSK the different phases encode information. A very simple variant of PSK is BPSK (Binary PSK). In this form of phase shift keying the signal only switches between two phases to encode information. An example of BPSK is shown by Figure 5. BPSK is used in the ISO/IEC 14443-B standard for card-to-reader communication. Here the initial phase $\phi = 0$ and represents a logic '1'. A phase change of 180° indicates a transition of this logical value. Phase changes are positioned at the edges of an ETU.

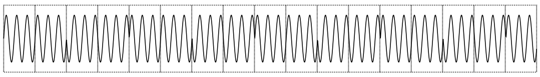

FIGURE 5 - PHASE SHIFT KEYING

III. GHOST

At the time of its development, there were no open source tools for the analysis of RFID protocols. The available commercial testing equipment was very expensive and focused on the verification of existing RFID standards and certification of compatible products. The Ghost, instead, was designed with the focus on security and to be capable of violating those standards by sending malicious messages, also known as protocol fuzzing. The hardware of the Ghost was designed by Peter Dolron, member of our TechnoCentrum[2].

3.1 Hardware

The Ghost is a portable and inexpensive open-design ISO/IEC 14443-A RFID tag emulator. It has the size of a matchbox and the total cost of the hardware components is approximately €40. It can emulate a tag and it is fully programable (e.g., you can emulate a custom UID). The Ghost is controlled by a PIC18F4620 microcontroller that runs custom firmware. This firmware controls the transmission layer which is used for communication. It is equipped with an RS232 serial interface that can be used to log eavesdropped transactions and to update the internal configuration. The Ghost can work standalone as it is powered by a 9V battery.

3.2 Software

The hardware of the Ghost is very simple and limited. The embedded firmware is responsible for the signal (de-)modulation and processing. This provides great flexibility but programming it is a tedious and time-consuming task. Additionally, we have developed the remote application RfidSpy to configure hardware settings and to process captured data frames in real-time. This application has a graphical user interface and runs under Microsoft Windows. With this software it is possible to change the UID, capture data and define custom responses to certain messages. Using the eavesdrop functionality it is possible to eavesdrop frames transmitted by the reader. The man-in-the-middle mode relays all communication to the PC. Finally, the emulation mode mimics the behavior of simple (i.e., without encryption) tags like the Mifare Ultralight.

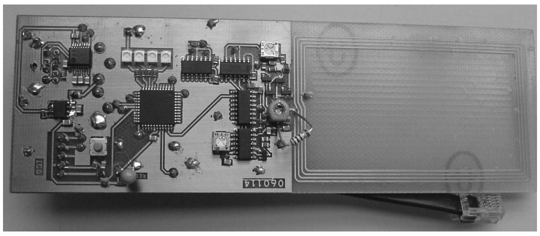

FIGURE 6 - THE GHOST

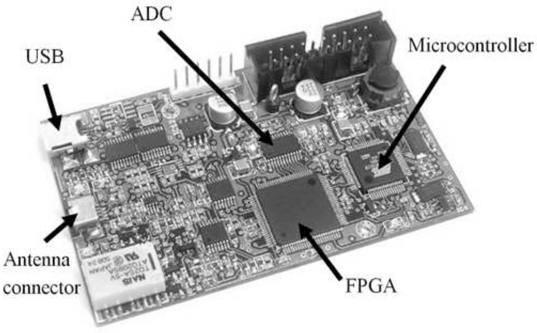

FIGURE 7 - THE PROXMARK III

3.3 Capabilities

The Ghost is a tag emulator which has two modes of operation.

3.3.1 Eavesdropping mode

In this mode, the Ghost eavesdrops frames that are sent by a reader. The Ghost does not respond to any frame, in this way it will not interfere with active transactions that are communicated between a reader and a genuine tag. The Ghost cannot demodulate the Manchester signal from a tag, it is only capable to understand the messages sent by the reader. If the protocol is not session-dependent, it is possible to replay the eavesdropped reader frames (using your own reader) to gather the missing answers from the tag. If there was a cryptographic challenge during the communication, the replay of the reader frames would not be useful.

3.3.2 Emulation mode

This mode transforms the Ghost into a lightweight tag. The user can supply a UID which is used in the anti-collision. Additionally, the user can provide some answers to predefined reader messages. Then, the Ghost simulates the responses during a replay attack. After its configuration, the Ghost can be disconnected from the computer and used as a standalone device, in this way it is compact and easy to hide.

Several access control and transport ticketing systems use constant identifiers, like serial numbers, for authentication. We have performed a replay attack [5–7] for low-cost disposable tickets used in public transport systems like the OV-chipkaart[3]. Every off-line system that uses non-encrypted memory cards like the Mifare Ultralight is vulnerable to such attacks.

IV. PROXMARK

The Proxmark III hardware has been developed by Jonathan Westhues [4]. The Proxmark III, shown in Figure 7, replaces its predecessors and introduces a high level of flexibility in both signal processing and protocol implementation. It is additionally equipped with a Field Programmable Gate Array (FPGA) which is mainly responsible for the low-level signal processing and allows to set up multiple signal processing schemes. In general, when we speak about the Proxmark, we refer to this latest version.

The hardware design and firmware of this latest version is in the public domain since May 2007 under the General Public License. The device costs around €200 and since the schematics are online, it can be ordered through any local printed circuit board (PCB) supplier. Although, most assembled Proxmark devices are sold by one of the main suppliers: Rysc Corp.[5], GeZhi Electronic Corp. Ltd.[6] and hackable-devices[7]. The following websites contain all the information that is required to assemble, compile, flash, use and develop new features for the Proxmark.

[2] http://www.ru.nl/fnwi/technocentrum/

[3] http://www.ov-chipkaart.nl

[4] http://cq.cx/proxmark3.pl

[5] http://www.proxmark3.com

[6] http://www.xfpga.com

[7] http://www.hackable-devices.org

- `http://cq.cx/proxmark3.pl` The first website about the Proxmark device, created by Jonathan Westhues in 2007. Jonathan made the project free to use and published all the necessary designs and source codes. Five years later, already more than a thousand Proxmarks were sold for extensive RFID protocol and security research.

- `http://www.proxmark.org` Contains a lot of information about new RFID modulation, encoding and protocols that were added the last years. This website hosts the main community forum, which is currently used by more than 3000 members. This forum answers all frequently asked questions concerning the Proxmark, but also contains various topics about microcontroller and FPGA development.

- `http://proxmark3.googlecode.com` This is the development website which hosts the most recent subversion (SVN) repository. Only in 2012 there are already 26 active committers, who regularly fix problems and contribute new features to the Proxmark firmware. The website also hosts a small wiki that contains a manual for using the Proxmark device. Most features and commands are explained in detail, backed up by several output examples and pictures.

4.1 Hardware

The Proxmark III supports both low (125 kHz-134 kHz) and high frequency (13.56 MHz) signal processing. This is achieved by two parallel antenna circuits that can be used independently. Both circuits are connected to a 4-pin Hirose connector to connect an external loop antenna. When the Proxmark is in *reader mode* it drives the antenna coils with the appropriate frequency. This is unnecessary when the Proxmark works in *eavesdropping mode* or in *card emulation mode* because then the electromagnetic field is generated by the reader. The signal from the antenna is routed through the FPGA after it has been digitized by an 8-bit Analog-to-Digital Converter (ADC). After some filtering, the FPGA relays the necessary information to perform the decoding of the signal to the microcontroller. This prevents the microcontroller from being overloaded with signal data. An FPGA has a great advantage over a normal microcontroller in the sense that it emulates hardware. A hardware description can be compiled and flashed into an FPGA. Basic arithmetic operations can be performed in parallel and faster than in a microcontroller. An FPGA is of course slower than a hardware implementation but pure hardware lacks flexibility.

4.1.1 FPGA implementation

We will now discuss the FPGA implementation of the ISO/IEC 14443-A standard in the Proxmark. This is implemented in the `hi_iso14443a` module. This module is implemented in Verilog, a hardware description language. In short, the FPGA is used for the modulation and demodulation part of the digital signal processing (DSP). We implemented the ASK modulation scheme on the FPGA and the Modified Miller and Manchester encoding schemes at the microcontroller. The FPGA samples at a clock speed of 13.56 MHz and sends the bits that describe the transitions of the modulated signal to the microcontroller. These bits are sent at a rate of 8 samples per period (847.5 kbps) for the reader-to-card signal. The FPGA code defines several modules that all make use of the same interfaces. A multiplexer switches between the FPGA modules that cover the high and low frequency processing. Every module runs with different parameters that are specific for sending, receiving and eavesdropping. The following modules (operating modes) are implemented for ISO/IEC 14443-A. *Eavesdroppping* (Mode 0); Samples the reader-to-card as well as card-to-reader communication at 1.7 Mbps. *Card listening* (Mode 1); Samples reader communication at 847.5 kbps. *Card transmitting* (Mode 2); Modulates card-to-reader communication and generates a subcarrier. *Reader listening* (Mode 3); Samples card-to-reader communication at 847.5 kbps and generates a carrier wave. *Reader transmitting* (Mode 4); Modulates reader-to-card communication and generates a carrier wave.

The Proxmark switches seamlessly between these operating modes. This is especially important when switching between transmitting and listening in the reader simulation modes (mode 3 and 4). If the Proxmark stops generating a field for a short moment this will result in the card losing its power and thus its current state.

mod_type	**ssp_clk**	**pwr_hi**	**pwr_oe4**
3’b000 (0)	clk2	0	0
3’b001 (1)	clk3	0	0
3’b010 (2)	clk3	0	clk3 & mod_sig_coil
3’b011 (3)	clk3	clk1 & $\overline{\texttt{mod\_sig\_coil}}$	0
3’b100 (4)	clk3	clk1	0

Where `clk1 = 13.56 MHz, clk2 = 1.695 MHz, clk3 = 847.5 KHz`

FIGURE 9 - OVERVIEW OF THE INTERFACES

4.1.2 Analog-to-Digital Converter

The Analog-to-Digital Converter (ADC) transfers the analog input voltage of the antenna to a digital representation. The Proxmark uses an 8-bit ADC (TLC5540) which divides the analog input into 256 output steps. Two reference voltages on input pins `REFB` and `REFT` correspond to the bottom and top input of the input range, respectively. This input range is divided in equally sized digital steps.

The analog input signal is connected to the `ANALOG-IN` pin of the ADC. The digitized output signal is put onto eight parallel output lines (`D1-D8`) that connect the ADC and FPGA. These lines are clearly visible on the Proxmark circuit board and are shown as bold lines in Figure 8.

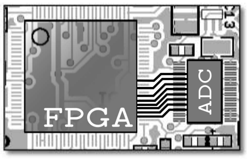

FIGURE 8 - ADC WIRING

4.1.3 FPGA interfaces

We will now discuss the most important input and output interfaces that are defined on the FPGA. A 13.56 MHz crystal is used for the high frequency FPGA modes. Once every period of the carrier wave, the FPGA evaluates the digital output of the ADC and processes this information into samples that are sent out to the ARM. Furthermore, the FPGA has two outputs (`pwr_lo` and `pwr_hi`) that are used to generate a low frequency or high frequency field. Apart from the carrier signal, the FPGA also controls four outputs that determine the level of load modulation on the signal. This comes into use when emulating a transponder. Figure 9 gives a quick overview of the signals on the most important output pins for the different FPGA modes.

4.1.4 Decoding and sampling reader-to-card communication

Decoding reader communication is easier compared to decoding card communication since the reader controls the field and can easily introduce signal changes. A card can only induce small changes in the reader field. The ISO/IEC 14443-A standard defines a default bit rate of $f_c/128 \approx 106$ kbit/s with $f_c = 13.56$ MHz. This means that one bit is transmitted every 128 clock cycles of the carrier wave. Recall from Section 2.1 that the reader-to-card communication for ISO/IEC 14443-A is encoded by Modified Miller. In this encoding scheme the signal is high at default. Information is encoded by dropping the reader field for very short periods at certain time intervals. These drops last $\frac{1}{4} \cdot 128 = 32$ clock cycles and occur at most once per bit period of 128 cycles. Following the Nyquist theorem we sample at twice the frequency of our encoded signal. This means that the FPGA samples the reader-to-card communication every 16 cycles. These samples are sent in binary format to the micorcontroller. The following bit string is an example of this communication from the FPGA to the microcontroller.

$$\ldots 1111\ \overbrace{00111111}^{\texttt{SOF}}\ \overbrace{00111111}^{0}\ \overbrace{11110011}^{1}\ \overbrace{11110011}^{1}\ \overbrace{11111111}^{0}$$
$$\underbrace{00111111}_{0}\ \underbrace{11110011}_{1}\ \underbrace{11111111}_{0}\ \underbrace{0011111111111111}_{\texttt{EOF}}\ 1111\ldots$$

This bit string encodes the *REQA* command[8] which is sent out by the

[8] The Request Type A command is used to select a card that (partly) supports ISO/IEC 14443-A.

reader continuously when it is polling for new cards. It consists of 7 bits (a so-called *short frame*) and is like all reader frames enclosed by a Start-of-Frame (SOF) and an End-of-Frame (EOF).

4.1.5 Decoding and sampling card-to-reader communication

In case of eavesdropping or reader simulation the Proxmark demodulates card signals. In order to do this, a more refined method is needed compared to the reader decoding. The main reason for this is that the card is powered by the reader field. It communicates to the reader by generating a subcarrier. This subcarrier affects the power level that is received by the Proxmark antenna. Although this induces changes in the power level, the changes are far more subtle and thus harder to detect. It is for this reason that reader communication can be eavesdropped at much larger distances than card communication. Despite the weaker signal of the card, the default bit rate is also 106 kbit/s. The communication is Manchester encoded as described in Section 2.2. During communication, half the bit periods have a modulated subcarrier, and half the bit periods do not contain any modulation. These periods last 64 cycles of the carrier wave and are twice as long as the signal drops in reader-to-card communication. However, the power variations are very small and the positioning of the card in the reader field highly influences the signal impedance. Therefore, a standard static threshold technique is not suitable in this situation. We use an *adaptive progressive thresholding* technique instead. It is a real-time problem where, at every period of the 13.56 MHz wave, the FPGA has to decide whether a transition took place or not. It is not feasible to reconsider earlier points in time. The thresholding technique that we use is adaptive since a variable threshold value is used at different points in time. Furthermore, it is progressive since it scans the digital signal for extremes and adjusts the thresholds according to these extremes.

The FPGA implementation averages the ADC output over 16 periods of the carrier wave. When the difference between two consecutive averages exceeds a threshold value it is counted as a transition. After every evaluation, the threshold value is updated to the difference between the last two averages. This corrects automatically for a too sensitive threshold value. On the other hand, a too insensitive threshold value is prevented by automatically resetting it to its initial value after a period without transitions. Concretely, it is impossible to have more than one bit period (8 samples) without transitions according to the ISO/IEC 14443-A standard. In our FPGA code we take some additional margin into account and reset the threshold after 16 identical samples. The following bit string is an example of what is send from the FPGA to the microcontroller.

$$\ldots 0000\,\underbrace{11110000}_{\text{SOF}}\,\underbrace{00001111}_{0}\,\underbrace{11110000}_{1}\,\underbrace{00001111}_{0}\,\underbrace{11110000}_{1}\,\underbrace{11111111}_{\text{EOF}}\,1100\ldots$$

This bit string encodes the acknowledgement (*ACK*) of a Mifare Classic card to some reader command. It consists of 4 bits and is like all card frames enclosed by a SOF and an EOF.

4.1.6 Microcontroller

The microcontroller is responsible for the protocol part. It receives the digital signal from the FPGA and decodes it. The decoded signal can just be copied to a buffer in the EEPROM memory. Additionally, an answer to an incoming message can be programmed to be sent immediately, communicating this to the FPGA which then modulates the appropriate signal.

4.2 Extending protocol support

The Proxmark can operate in three different mores: eavesdropping mode; card emulation mode; and reader mode. It is possible to use the Proxmark for very different modulation schemes and protocols as long as they are in the supported frequency range. Some well known protocols and modulation schemes are already available like the ISO/IEC 14443-A implementation that we discussed earlier. There are some requirements to implement the mentioned different modes for new protocols. First, an underlying physical layer is needed that takes care of the Digital Signal Processing (DSP). This is mostly done at the FPGA. Then, the encoding and decoding schemes should be implemented as functions on the microcontroller. Finally, the client should be able to call these functions in order to emulate a tag or reader and display the results. The hardware design that can be flashed into the FPGA is written in Verilog. Verilog is a hardware description language which allows to describe a hardware design in a C-style syntax.

The client application (originally written by Jonathan Westhues and later improved by the Proxmark community) connects to the Proxmark via the standard HID USB protocol. Since the microcontroller polls for new USB packets, it is not possible to stream the retrieved samples to the PC in real-time. When the microcontroller retrieves a command from the client it runs this command and stores any resulting messages in its memory buffer. Later, the client needs to send a new command to receive the data from this buffer.

4.3 Capabilities

This section describes the strong points of the Proxmark. In general, most of its flexibility comes from the software defined protocol stack that is implemented on the FPGA and the microcontroller.

4.3.1 Supporting user defined modulation

It is possible to develop new modulation and protocol interpreters by only adjusting the software. It supports Amplitude Shift Keying (ASK), Binary Phase Shift Keying (BPSK) and On-Off Keying (OOK) for two major RFID frequencies 125-135 kHz and 13.56 MHz. We have initiated a small but steadily growing community[9] of Proxmark developers. By now there is support for various widely used RFID tags like SRI512, Mifare, ICODE SLI, EM4200, Nedap, iClass, Indala, HID Prox. Using these open source examples it is rather easy to add support for new RFID modulations and protocols.

4.3.2 Eavesdropping modulation schemes simultaneously

The Proxmark is capable of simultaneously eavesdropping Modified Miller and Manchester modulated communication. As both modulation techniques are part of one ISO/IEC 14443-A session, it is possible to get both reader and tag messages in one capture. To analyze unknown RFID protocols it is very convenient to get a full trace of the transaction. This feature was of great use in the attack proposed in [33, 58] to retrieve the 64 bits keystream from one authentication session to construct the optimized table attack exploiting linear combinations from the CRYPTO1 cipher.

4.3.3 Mifare Classic emulation

We have implemented the CRYPTO1 cipher in the Proxmark. This enables emulation of Mifare Classic from both tag and reader side. This feature is of great aid while assessing the security of integrated systems like access control and public transport systems. It is especially useful to assess the identity fraud detection mechanisms of such systems. We successfully attacked several systems using this feature and pointed out the weak spots to the system integrators.

4.3.4 Timing information

Since the demodulation is done in real-time by the FPGA, the Proxmark provides accurate timing information. This is very useful for the study of potential side channel attacks. Pseudo Random Number Generators (PRNG) often rely on timing. When an attacker can observe the related time in such a manner that the generator becomes predictable, a security scheme can not rely anymore on the assumption of an unpredictable PRNG. This feature can help to discover relations between the timing information and the numbers generated by the PRNG of a tag. This feature was used to recover the PRNG timing relation of a Mifare Classic tags in [33, 45]. This relation was also exploited later in [47].

V. LIBNFC

Near Field Communication (NFC) is an open platform technology that provide wireless communication between devices that are compatible

[9] http://www.proxmark.org/forum

with the ISO/IEC 18092 [59] and ISO/IEC 21481 [60] standard. It was proposed as a unifying successor for the available RFID technologies. The NFC standard wraps older RFID modulations like the ISO/IEC 14443 [10], ISO/IEC 15693 [11] and JIS X 6319-4 [12]. There are a number of devices compatible with NFC technology like GSM phones and other peripheral computer equipment[10].

5.1 Hardware

The NFC Controllers used by major manufacturers (Nokia, NXP, Sony, etc.) are based on the NXP 80C51 microprocessor as described in [61]. These chips are pre-configured with an embedded firmware and distributed as controller IC's PN531, PN532 and PN533. Most mobile phones, POS terminals and desktop NFC devices are using this controller. They are cheap, reliable, support advanced features and are very easy to use. The chips can be interfaced through a UART, SPI, I^2C and USB connection. The controller just needs to be connected to the antenna and is ready to connect directly to a computer using the USB interface. In practice, manufactures often choose to put a IC in between the controller and the computer that performs some proprietary functionality. The IC might run an embedded firmware and run standalone. Since one chip is enough to support all NFC communication, this brings the average price of an NFC device down to €30. Together with MicroBuilder SARL[11] we designed and build an open hardware board that is fully supported by Libnfc. The schematics, components and hardware design is completely open and released under the Creative Commons Attribution-Share Alike 3.0 Unported License. A picture of this reader is shown in Fig 10.

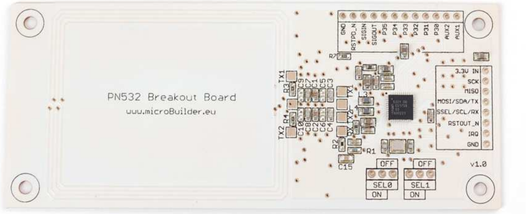

FIGURE 10 - OPEN DESIGN LIBNFC READER

There are three different communication interfaces available, which can be configured with two jumpers. It supports the Serial Peripheral Interface (SPI) bus, Inter-Integrated Circuit (I^2C) and Universal Asynchronous Receiver/Transmitter (UART). The SPI and I^2C are dedicated communication lines with their own transmission clock. Both require four wires and are often deployed in embedded devices that use a master-slave controlling environment. The UART interface is slower, but it requires only two wires and is since decades a widely deployed communication channel. The UART interface is convenient for backwards compatibility with regular personal computers and older embedded devices. Because of the low production costs of this Libnfc device, the connecting capabilities and the ease of configuring, it is eminently for quickly prototyping of new NFC designs.

5.2 Software

There are various commercial SDK's[12][13] available for NFC. They are hardware dependant and use proprietary software. They are not platform independent and give no transparency over their functionality. Most NFC readers are using the PN53X chipset. These chips have an extensive firmware which supports lots of features. These features are invoked by a simple instruction set, the TAMA language. Most SDK developers allow TAMA instructions to bypass their own layer and are sent directly to the NFC controller. Libnfc makes use of this property. Since the TAMA language is the same for all commercial NFC readers, it is possible to have a library that supports NFC readers from different vendors. The source code from Libnfc is portable to different platforms. So far, it has been tested on Linux, Mac OS X and Windows. OpenPICC2[14] shows that it is possible to compile Libnfc into an embedded firmware so that it can be used in a standalone device.

The API of Libnfc consists of three sections. The first part covers the general functionality, like the connection and configuration of an NFC device. The second part focuses on the NFC device as an RFID reader (initiator in NFC terms). The automatic initialization of various tags can be invoked by just one function call. The firmware of the NFC Controller will handle the modulation, startup and selection process. However, this can also be done manually using the raw-frames transceiver functionality. It supplies the feature to send arbitrary parity bits, optional CRC bytes and custom frame-lengths. This gives the developer great control over the frames transmitted by the NFC device. The last part focuses on tag (target in NFC terms) emulation. It supports comparable features with the Initiator part. Besides, it is possible to let the NFC controller emulate a tag automatically. These features enables developers to prototype proposals from the literature [62, 63] that rely on the NFC interface as communication channel.

5.3 Capabilities

The library contains different drivers for NFC devices available from most major manufacturers. Developers using Libnfc might abstract from which device or interface (USB, SERIAL, I^2C) to the NFC controller is being used. This makes the development vendor independent and thereby prevents vendor lock-ins. There are five example utilities included in the Libnfc package. Every utility demonstrates a powerful feature of Libnfc using only a few lines of code. These utilities can be used as kick start for developing new software. In this section there is a brief description of each of them.

5.3.1 Anticol

Anti-collision demonstration tool for ISO/IEC 14443-A tags allows sending custom of anti-collision frames. The first frame must be a short frame which is only 7 bits long. Commercial SDK's often don't support a feature to send frames that are not a multiple of 8 bits (1 byte) long. The developer has to rely on closed proprietary software and should hope it does not contain vulnerabilities during the anti-collision phase. Performing the anti-collision using custom frames could protect against a malicious tag that, for example, violates the standard by sending frames with unsupported lengths.

5.3.2 Emulate

Tag emulation is one of the main added features NFC. To avoid abuse of existing systems, manufacturers of the NFC controller intentionally did not support emulation of custom UID numbers. The emulate tool demonstrates that this can still be done using transmission of raw-frames. It is necessary to respond in time for the anti-collision protocol. The USB interface introduces response delays, but an embedded microprocessor is fast enough to emulate a tag with any UID. This makes it a serious thread for security systems that rely only on the uniqueness of the UID.

5.3.3 List

The list utility attempts to select available tags in the field. The NFC controller is used to perform the selection procedure. This is different for each modulation type. It tries to find a ISO/IEC 14443 type A, type B, Felica or Jewel Topaz tags. This tool demonstrates that it is possible to setup a simple NFC system using less than 10 lines of code.

5.3.4 MFtool

The Mifare Classic tag is one of the most widely used RFID tags. The firmware in the NFC controller supports authenticating, reading and writing to/from Mifare Classic tags. This tool demonstrate the speed of this library and its ease-of-use. It possible to read and write the complete content of a Mifare Classic 4K tag within 1 second.

[10] http://www.libnfc.org/hardware/compatibility
[11] http://www.microbuilder.eu/Projects/PN532
[12] http://www.stollmann.de/en/stacks/nfc/
[13] http://www.acs.com.hk/acr122-sdk.php
[14] http://www.libnfc.org/hardware/devices/openpicc2

5.3.5 Relay

The relay utility demonstrates a relay attack. For this it requires two NFC devices. One will emulate an ISO/IEC 14443 type A tag, while the second device will act as a reader. The genuine tag can be placed on the 2nd reader and the tag emulator can be placed close to the original reader. All communication is now relayed and shown in the screen on real-time.

VI. Conclusions

We have presented three tools that should satisfy the needs of most RFID researchers. Those having an NFC reader can already start scrutinizing several RFID protocols using Libnfc. Those who need precise timing information or quick response times should get their hands on a Proxmark III. This is by far the most powerful of the tools presented here, but its operation requires some training. Finally, if you need an inexpensive and portable standalone tag emulator and you only need modest computing power, you can build a Ghost. All these tools are available under the GNU General Public License so feel free to contribute and expand the community.

References

[1] MIFARE Ultralight, MF0ICU1. Functional specification, February 2008. NXP Semiconductors.

[2] Tag-it hf-i transponder IC, TMS37112. Public Reference Guide, July 2005. Texas Instruments Incorporated.

[3] 512 bit read/write, ISO15693 standard compliant contactless rw identification device, EM4133. Public Datasheet, May 2008. EM Microelectronic-Marin SA.

[4] 13.56 mhz short-range contactless memory chip with 512-bit EEPROM and anticollision functions, SRT512. Public Datasheet, September 2011. ST Microelectronics.

[5] Roel Verdult. Proof of concept, cloning the OV-chip card. Technical report, Radboud University Nijmegen, 2008.

[6] Roel Verdult. Security analysis of RFID tags. Master's thesis, Radboud University Nijmegen, 2008.

[7] Gerhard de Koning Gans. Analysis of the MIFARE Classic used in the OV-chipkaart project. Master's thesis, Radboud University Nijmegen, 2008.

[8] Radio frequency identification of animals – code structure (ISO/IEC 11784), 1994. International Organization for Standardization (ISO).

[9] Radio frequency identification of animals – technical concept (ISO/IEC 11785), 1996. International Organization for Standardization (ISO).

[10] Identification cards – contactless integrated circuit cards – proximity cards (ISO/IEC 14443), 2001. International Organization for Standardization (ISO).

[11] Identification cards – contactless integrated circuit(s) cards – vicinity cards (ISO/IEC 15693), 2000. International Organization for Standardization (ISO).

[12] Specification of implementation for integrated circuit(s) cards (JICSAP/JSA JIS X 6319), 2005. Japan IC Card System Application Council (JICSAP).

[13] MIFARE Classic 1k, MF1ICS50. Public product data sheet, July 1998. Philips Semiconductors.

[14] KeeLoq crypto read/write transponder module, HCS410/WM. Product Datasheet, Jan 2001. Microchip Technology Incorporated.

[15] 13.56 mhz short range contactless memory chip with 4096 bit EEPROM, anti-collision functions and anti-clone functions, SRIX4K. Preliminary Product Datasheet, May 2002. ST Microelectronics.

[16] 125khz crypto read/write contactless identification device, EM4170. Product Datasheet, Mar 2002. EM Microelectronic-Marin SA.

[17] Radio frequency identification systems – digital signature transponder plus, DST+. Product Specification, July 2004. Texas Instruments Incorporated.

[18] PicoPass 2KS. Product Datasheet, Nov 2004. Inside Contactless.

[19] 13.56 MHz contactless iClass card. Product Features and Specifications, October 2008. HID Global.

[20] CryptoRF specification, AT88SCxxxxCRF. Product Datasheet, March 2009. Atmel Corporation.

[21] Transponder IC, Hitag2. Product Data Sheet, Nov 2010. NXP Semiconductors.

[22] Auguste Kerckhoffs. La cryptographie militaire. *Journal des Sciences Militaires*, 9(1):5–38, 1883.

[23] Norman D. Jorstad and Landgrave T. Smith. Cryptographic algorithm metrics. In *20th National Information Systems Security Conference*. National Institute of Standards and Technology (NIST), 1997.

[24] Jovan Dj. Golić. Cryptanalysis of alleged A5 stream cipher. In *16th International Conference on the Theory and Application of Cryptographic Techniques, Advances in Cryptology (EUROCRYPT 1997)*, volume 1233 of *Lecture Notes in Computer Science*, pages 239–255. Springer-Verlag, 1997.

[25] John Kelsey, Bruce Schneier, and David Wagner. Related-key cryptanalysis of 3-WAY, Biham-DES, CAST, DES-X, NewDES, RC2, and TEA. In *1st International Conference on Information and Communications Security (ICICS 1997)*, volume 1334 of *Lecture Notes in Computer Science*, pages 233–246. Springer-Verlag, 1997.

[26] Jovan Dj. Golić. Linear statistical weakness of alleged RC4 keystream generator. In *16th International Conference on the Theory and Application of Cryptographic Techniques, Advances in Cryptology (EUROCRYPT 1997)*, volume 1233 of *Lecture Notes in Computer Science*, pages 226–238. Springer-Verlag, 1997.

[27] David Wagner, Leone Simpson, Ed Dawson, John Kelsey, William Millan, and Bruce Schneier. Cryptanalysis of ORYX. In *5th International Workshop on Selected Areas in Cryptography (SAC 1998)*, volume 1556 of *Lecture Notes in Computer Science*, pages 631–631. Springer-Verlag, 1999.

[28] Frank A. Stevenson. Cryptanalysis of contents scrambling system (CCS), November 1999.

[29] Eli Biham, Alex Biryukov, and Adi Shamir. Cryptanalysis of Skipjack reduced to 31 rounds using impossible differentials. In *18th International Conference on the Theory and Application of Cryptographic Techniques, Advances in Cryptology (EUROCRYPT 1999)*, volume 1592 of *Lecture Notes in Computer Science*, pages 12–23. Springer-Verlag, 1999.

[30] M. Hermelin and K. Nyberg. Correlation properties of the Bluetooth combiner. *2nd Information Security and Cryptology (ICISC 1999)*, 1787:17–29, 2000.

[31] Stephen C. Bono, Matthew Green, Adam Stubblefield, Ari Juels, Aviel D. Rubin, and Michael Szydlo. Security analysis of a cryptographically-enabled RFID device. In *14th USENIX Security Symposium (USENIX Security 2005)*, pages 1–16. USENIX Association, 2005.

[32] Andrey Bogdanov. Linear slide attacks on the KeeLoq block cipher. In *Information Security and Cryptology (INSCRYPT 2007)*, volume 4990 of *Lecture Notes in Computer Science*, pages 66–80. Springer, 2007.

[33] Flavio D. Garcia, Gerhard de Koning Gans, Ruben Muijrers, Peter van Rossum, Roel Verdult, Ronny Wichers Schreur, and Bart Jacobs. Dismantling MIFARE Classic. In *13th European Symposium on Research in Computer Security (ESORICS 2008)*, volume 5283 of *Lecture Notes in Computer Science*, pages 97–114. Springer-Verlag, 2008.

[34] Stefan Lucks, Andreas Schuler, Erik Tews, Ralf-Philipp Weinmann, and Matthias Wenzel. Attacks on the DECT authentication mechanisms. In *9th Cryptographers' Track at the RSA*

Conference (CT-RSA 2009), volume 5473 of *Lecture Notes in Computer Science*, pages 48–65. Springer-Verlag, 2009.

[35] Nicolas T. Courtois, Sean O'Neil, and Jean-Jacques Quisquater. Practical algebraic attacks on the Hitag2 stream cipher. In *12th Information Security Conference (ISC 2009)*, volume 5735 of *Lecture Notes in Computer Science*, pages 167–176. Springer-Verlag, 2009.

[36] Flavio D. Garcia, Peter van Rossum, Roel Verdult, and Ronny Wichers Schreur. Dismantling SecureMemory, CryptoMemory and CryptoRF. In *17th ACM Conference on Computer and Communications Security (CCS 2010)*, pages 250–259. ACM/SIGSAC, 2010.

[37] Henryk Plötz and Karsten Nohl. Peeling away layers of an RFID security system. In *16th International Conference on Financial Cryptography and Data Security (FC 2012)*, volume 7035 of *Lecture Notes in Computer Science*, pages 205–219. Springer-Verlag, 2012.

[38] Benedikt Driessen, Ralf Hund, Carsten Willems, Carsten Paar, and Thorsten Holz. Don't trust satellite phones: A security analysis of two satphone standards. In *33rd IEEE Symposium on Security and Privacy (S&P 2012)*, pages 128–142. IEEE Computer Society, 2012.

[39] Flavio D. Garcia, Gerhard de Koning Gans, Roel Verdult, and Milosch Meriac. Dismantling iClass and iClass Elite. In *17th European Symposium on Research in Computer Security (ESORICS 2012)*, Lecture Notes in Computer Science. Springer-Verlag, 2012.

[40] Filippo Gandino, Renato Ferrero, Bartolomeo Montrucchio, and Maurizio Rebaudengo. Probabilistic DCS: An RFID reader-to-reader anti-collision protocol. *Journal of Network and Computer Applications*, 34(3):821–832, 2011.

[41] Renato Ferrero, Filippo Gandino, Bartolomeo Montrucchio, and Maurizio Rebaudengo. Fair anti-collision protocol in dense rfid networks. In *3rd International EURASIP Workshop on RFID Technology (EURASIP-RFID 2010)*, pages 101–105. IEEE Computer Society, 2010.

[42] Gerhard de Koning Gans and Flavio D. Garcia. Towards a practical solution to the RFID desynchronization problem. In *6th Workshop on RFID Security (RFIDSec 2010)*, volume 6370 of *Lecture Notes in Computer Science*, pages 203–219. Springer-Verlag, 2010.

[43] Flavio D. Garcia and Peter van Rossum. Modeling privacy for off-line RFID systems. In *9th Smart Card Research and Advanced Applications (CARDIS 2010)*, volume 6035 of *Lecture Notes in Computer Science*, pages 194–208. Springer-Verlag, 2010.

[44] Ronny Wichers Schreur, Peter van Rossum, Flavio D. Garcia, Wouter Teepe, Jaap-Henk Hoepman, Bart Jacobs, Gerhard de Koning Gans, Roel Verdult, Ruben Muijrers, Ravindra Kali, and Vinesh Kali. Security flaw in MIFARE Classic. *Press release, Digital Security group, Radboud University Nijmegen, The Netherlands*, March 2008.

[45] Gerhard de Koning Gans, Jaap-Henk Hoepman, and Flavio D. Garcia. A practical attack on the MIFARE Classic. In *8th Smart Card Research and Advanced Applications Conference (CARDIS 2008)*, volume 5189 of *Lecture Notes in Computer Science*, pages 267–282. Springer-Verlag, 2008.

[46] Karsten Nohl, David Evans, Starbug, and Henryk Plötz. Reverse engineering a cryptographic RFID tag. In *17th USENIX Security Symposium (USENIX Security 2008)*, pages 185–193. USENIX Association, 2008.

[47] Flavio D. Garcia, Peter van Rossum, Roel Verdult, and Ronny Wichers Schreur. Wirelessly pickpocketing a MIFARE Classic card. In *30th IEEE Symposium on Security and Privacy (S&P 2009)*, pages 3–15. IEEE Computer Society, 2009.

[48] Nicolas T. Courtois. The dark side of security by obscurity - and cloning MIFARE Classic rail and building passes, anywhere, anytime. In *4th International Conference on Security and Cryptography (SECRYPT 2009)*, pages 331–338. INSTICC Press, 2009.

[49] Alex Biryukov, Ilya Kizhvatov, and Bin Zhang. Cryptanalysis of the Atmel cipher in SecureMemory, CryptoMemory and CryptoRF. In *9th Applied Cryptography and Network Security (ACNS 2011)*, volume 6715 of *Lecture Notes in Computer Science*, pages 91–109. Springer-Verlag, 2011.

[50] Josep Balasch, Benedikt Gierlichs, Roel Verdult, Lejla Batina, and Ingrid Verbauwhede. Power analysis of Atmel CryptoMemory - recovering keys from secure EEPROMs. In *12th Cryptographers' Track at the RSA Conference (CT-RSA 2012)*, volume 7178 of *Lecture Notes in Computer Science*, pages 19–34. Springer-Verlag, 2012.

[51] Flavio D. Garcia, Gerhard de Koning Gans, and Roel Verdult. Exposing iClass key diversification. In *5th USENIX Workshop on Offensive Technologies (USENIX WOOT 2011)*, pages 128–136. USENIX Association, 2011.

[52] Roel Verdult, Flavio D. Garcia, and Josep Balasch. Gone in 360 seconds: Hijacking with Hitag2. In *21st USENIX Security Symposium (USENIX Security 2012)*. USENIX Association, 2012.

[53] Roel Verdult and François Kooman. Practical attacks on NFC enabled cell phones. In *3rd International Workshop on Near Field Communication (NFC 2011)*, pages 77–82. IEEE Computer Society, 2011.

[54] Melanie Rieback, Bruno Crispo, and Andrew Tanenbaum. RFID guardian: A battery-powered mobile device for RFID privacy management. In *10th Australasian Conference on Information Security and Privacy (ACISP 2005)*, volume 3574 of *Lecture Notes in Computer Science*, pages 184–194. Springer-Verlag, 2005.

[55] Omar Choudary. The Smart Card Detective: A Hand-Held EMV Interceptor. Master's thesis, University of Cambridge, 2010.

[56] Gerhard de Koning Gans and Joeri de Ruiter. The smartlogic tool: Analysing and testing smart card protocols. In *5th International Conference on Software Testing, Verification, and Validation*, pages 864–871. IEEE Computer Society, 2012.

[57] Arjan Blom, Gerhard de Koning Gans, Erik Poll, Joeri de Ruiter, and Roel Verdult. Designed to fail: A USB-connected reader for online banking. In *17th Nordic Conference on Secure IT Systems (NordSec 2012)*, Lecture Notes in Computer Science. Springer-Verlag, 2012.

[58] Flavio D. Garcia, Gerhard de Koning Gans, and Roel Verdult. Tutorial: Proxmark, the swiss army knife for RFID security research. Technical report, Radboud University Nijmegen, 2012.

[59] Information technology – telecommunications and information exchange between systems – near field communication interface and protocol 1 (NFCIP-1) (ISO/IEC 18092), 2004. International Organization for Standardization (ISO).

[60] Information technology – telecommunications and information exchange between systems – near field communication interface and protocol 2 (NFCIP-2) (ISO/IEC 21481), 2005. International Organization for Standardization (ISO).

[61] Near field communication pn531-μc based transmission module. Short Form Specification, February 2004. NXP Semiconductors.

[62] Gauthier Van Damme, Karel M. Wouters, Hakan Karahan, and Bart Preneel. Offline NFC payments with electronic vouchers. In *1st ACM Workshop on Networking, Systems, and Applications on Mobile Handhelds (MobiHeld 2009)*, pages 25–30. ACM, 2009.

[63] Gergely. Alpár, Lejla Batina, and Roel Verdult. Using NFC phones for proving credentials. In *16th Measurement, Modelling, and Evaluation of Computing Systems and Dependability and Fault Tolerance (MMB&DFT 2012)*, volume 7201 of *Lecture Notes in Computer Science*, pages 317–330. Springer-Verlag, 2012.