On the Power of Active Relay Attacks using Custom-Made Proxies

Thomas Korak and Michael Hutter
Institute for Applied Information Processing and Communications (IAIK),
Graz University of Technology, Inffeldgasse 16a, 8010 Graz, Austria.
Email: {thomas.korak,michael.hutter}@iaik.tugraz.at

***Abstract*—A huge number of security-relevant systems nowadays use contactless smart cards. Such systems, like payment systems or access control systems, commonly use single-pass or mutual authentication protocols to proof the origin of the card holder. The application of relay attacks allows to circumvent this authentication process without needing to attack the implementation or protocol itself. Instead, the entire wireless communication is simply forwarded using a proxy and a mole allowing to relay messages over a large distance. In this paper, we present several relay attacks on an ISO/IEC 14443-based smart card implementing an AES challenge-response protocol. We highlight the strengths and weaknesses of two different proxy types: an NFC smart phone and a dedicated custom-made proxy device. First, we propose a "three-phones-in-the-middle" attack that allows to relay the communication over more than 360 feet (110 meters). Second, we present a custom-made proxy that solves major relay-attack restrictions that apply on almost all NFC smart phones, for example, cloning of the victim's UID, adaption of low-level protocol parameters, direct request for Waiting Time Extensions, or active modifications of the messages. Finally, we propose an attack that allows to induce single bit faults during the anticollision of the card which forces the reader to re-send or temporarily stall the communication which can be exploited by attacks to gain additional relay time.**

Keywords: Smart Cards, Radio-Frequency Identification (RFID), Relay Attacks, Man-in-the-Middle, Embedded Systems.

I. Introduction

Contactless smart cards based on the Radio Frequency Identification (RFID) or Near Field Communication (NFC) technology are widely used in various of applications nowadays. Typical applications, for example, are ticketing, access control, or cashless payment. The main reasons for the popularity of this technology compared to contact-based smart cards used decades ago are obvious: no direct contact to a reader device is required anymore which makes the handling easier and more comfortable. Major payment operators like Mastercard and Visa already provide contactless credit cards that use the wireless communication technology. These cards operate up to a distance of 10 centimeters according to the used ISO/IEC 14443 smart card standard. This fact implies also several threats as it was shown in the last years where RFID/NFC has been applied in more and more security-related applications. So-called *relay attacks* are one of many attacks that exploit the contactless communication technology. The main idea is to place a proxy device (or often referred to as *ghost*), which impersonates a victim's card, in close proximity to the reader. The proxy then forwards all messages to a mole (or often referred to as *leech*) that fakes an authentic reader to a victim's card. The distance between the proxy and the mole can thereby be as large as possible and as far as the response time is sufficiently short, e.g., Sportiello and Ciardulli recently demonstrated a successful relay attack over more than 300 miles [20]. Eddie Lee [2], as another example, successfully relayed the communication of contactless payment systems using two (low-cost) NFC smart phones over a WiFi relay channel. However, NFC-mobile phones as proxies or moles, as reported and used also in [2], [6], [20], lack in low relay times typically much larger than 10 milliseconds. Custom-made proxies using analog RFID-relay circuits, in contrast, are very fast (less than a few microseconds) but they are passive and do not allow modifications of the relay content.

In this paper, we present practical results of an active relay attack using a custom-made proxy device as a main contribution. The proxy uses a microcontroller which allows active modifications of relay content (as opposed to state of the art NFC phones). It has been assembled with low costs and can have a form factor much smaller than mobile phones which makes a detection of an attacker much harder because it can be simply hidden in wallets, for instance. With this proxy, low-level protocol parameters can be modified or added such as the unique ID (UID) or commands for Waiting Time Extensions (WTX). Besides this proxy, we provide the following novel contributions:

- We first pinpoint the advantages and disadvantages of NFC-based relay proxies compared to custom-made hardware. Custom proxies solve many relay-attack restrictions, e.g., cloning of the victim's UID, adaption of low-level ISO/IEC protocol parameters, direct request for Waiting Time Extensions, or modifications in the lower-level RFID protocols.
- We first present "three-phones-in-the-middle" attacks where we use one NFC phone to act as an access point for two other phones. Using this setup, we demonstrate a successful relay attack over a distance of more than 360 feet (110 meters).
- We compare the most relevant relay channels used for smart phones, i.e., the *Internet* (*WLAN*) and *Bluetooth* relay channels, and evaluate their performance regarding relay distance and speed. Practical results of performed relay attacks are given.

The work has been supported by the Austrian government through the research program SeCoS (project number 836628).

 DOI: http://dx.doi.org/10.1109/RFID.2014.6810722

- We introduce an ISO/IEC compliant way to extend the relay time during the anticollision loop of ISO/IEC 14443 A. By injecting bit collisions in the tree walking algorithm, an adversary is able to extend the relay time up to several seconds if needed. The proposed method is useful in cases where an unknown but constant UID of a card has to be relayed in a first pass or if an interleaved (challenge response) protocol is used in practice, as proposed by Feldhofer [4].
- Compared to the recent work of Francis et al. [5] we present a more effective relay attack by applying a custom-made proxy that is highly flexible. It allows more sophisticated relay attacks by custom parameterizations and optimizations on different communication layers, e.g., increasing the relay distance.

The remainder of the work is structured as follows. In Section II, an introduction to the ISO/IEC 14443 standard is given. Section III describes different setups for relay attacks in detail. In Section IV, we present the performed attacks. Section V discusses the results of the performed experiments and Section VI concludes the paper.

II. The ISO/IEC 14443 Timing Constraints

The ISO/IEC 14443 [10] is an international standard that defines all necessary parts to allow a contactless communication with identification cards. These cards can be smart cards, RFID transponders, or any other integrated circuits that are attached to an antenna are referred to as Proximity Integrated Circuit Cards (PICCs)[1] that communicate with a reader—the Proximity Coupling Device (PCD). The physical characteristics, power and signal interfaces, and communication protocols for both PICCs and PCDs are defined in four parts. Part one and two define the mechanical properties, dimensions, and modulation/demodulation types and the necessary coding methods. Part three, in particular important for this paper and the following terminology, specifies the initialization and anticollision process between the PICCs and the PCD. During this phase, all PICCs in the reading field of the PCD are getting selected by challenging them with a request command (defined as `REQA` or `REQB` in the standard types A and B, respectively). If there are more than one PICCs in the field, an anticollision loop is initiated that is used to detect and correctly select all PICCs in the proximity. We now list the most important parameters of the standard that are required for the attacks described in the following sections. The numbers given are valid for a bit rate of 106 kbps, for higher bit rates the numbers are slightly different.

The Frame Delay Time (FDT). The FDT is the time between two frames in opposite direction. During initialization and anticollision, the FDT from PCD to PICC has to be 91.15 μs (if the last bit of the PCD frame equals 1) or 86.43 μs (if the last bit of the PCD frame equals 0) in case of short frames[2]. For standard frames (higher-level commands), the FDT is calculated according to Equation 1 (last bit 1) and Equation 2 (last bit 0), respectively. f_c equals to the carrier frequency, i.e., 13.56 MHz.

The FDT from PICC to PCD has to be at least 86.4 μs, i.e.,

$$FDT = \frac{n \cdot 128 + 84}{f_c} \tag{1}$$

and

$$FDT = \frac{n \cdot 128 + 20}{f_c}, \tag{2}$$

where $n \geq 9$ (in case of a 106 kbit/s data rate).

The Frame Waiting Time (FWT). The FWT equals the maximum time the PCD has to wait for an answer of the PICC. This value can be set by the PICC in the Answer to Select (`ATS`) command using the according Frame Waiting Integer (FWI) value. FWI values in the range from 0 to 14 are supported which lead to FWT values between 302 μs and 4 989 ms (c.f. Equation 3).

$$FWT = \left(256 \cdot \frac{16}{f_c}\right) \cdot 2^{FWI} \tag{3}$$

Waiting Time Extension (WTX). If the PICC requires more time than the current FWT in order to process the received command from the PCD, it can request an extension of the time by sending a Waiting Time Extension (`WTX`) command to the PCD. The PCD has to answer with the same WTX command for confirmation. The maximum value the waiting time can be extended to is 4 949 ms. The WTX concept is useful for cryptographic calculations as they are typically very time consuming.

III. Setups for Relay Attacks and Related Work

There exist different ways and setups to perform relay attacks. Amongst the most straight-forward solutions is to use an NFC-enabled smart phone. The phone can act as a proxy device (by receiving and forwarding data from a PCD) and/or as a mole (acting as a PCD to a target PICC). Another solution is to use custom hardware to relay the communication. In the following, we describe these two setups and discuss advantages and disadvantages of their usage in practical relay attacks.

A. Relay using NFC-Enabled Smart Phones

According to NFC World [17], more than 160 smart phones that can be bought today support NFC (September 2013). There are further more than 20 smart phones that have been announced and will be shipped soon which will integrate NFC capabilities. There are only a few phones that do not have NFC, e.g., Apple's iPhone, the ZTE N880E, BlackBerry 9220, Samsung Galaxy S II, or the Android 2.3+ phones. By now there also exist some Japanese smart phones that are based on the very similar technology called "Osaifu Keitai" which uses the FeliCa[3] communication standard to implement mobile payment applications.

[1] Proximity-coupled cards operate in the electromagnetic near-field of a reader device that emits either a 125 kHz (LF) or 13.56 MHz (HF) carrier signal. The typical reading range of these systems is 10 cm.

[2] Short frames are used to initiate a communication, e.g., `REQA` or `WUPA` commands.

[3] FeliCa is an RFID communication standard specified by Sony. Mobile FeliCa is the technology for mobile payments developed in cooperation with Japan's phone operator NTT DoCoMo.

Next to NFC, common smart phones also support the use of other communication technologies such as WLAN, GSM, or Bluetooth. These technologies can be also used as relay channels as we will describe in the next section. Before this, we list the main advantages of these phones for performing relay attacks and also highlight their disadvantages and limitations.

Advantages. One of the most obvious advantages of smart phones is the *ease of use*. There exist many tools and software development environments to write own software to communicate with other devices in the proximity or even around the world (using WLAN or GSM/UMTS/LTE). The phones usually integrate hardware support for any communication technology that can be accessed by a specified API. That means that users can simply transfer messages over a communication channel (e.g., Bluetooth or WLAN) by calling a simple API function that is executed transparently to the application. Furthermore, smart phones are widely used and integrated in our community so that they do not attract much attention when they are applied in a relay attack. They are therefore harder to detect in practice compared to custom made proxy/mole devices that have a different and unknown form and shape.

Disadvantages. What has been described as a big advantage of NFC-enabled smart phones on the one hand, is one of the largest limitations on the other hand. Almost all smart phones integrate special hardware ICs for the individual communication technologies. These hardware modules can be used and accessed only by set of given interfaces. The use and even the modification of certain lower-level commands is highly limited and often not possible in practice. For instance, for most of the available smart phones, the Unique Identifier (UID) is constant (unalterable) or random (except of some constant manufactory-dependent bytes). Practical relay attacks using mobile phones therefore often assume that the UID is not going to be checked by the RFID system (because the UID can actually not be cloned and relayed by the proxy). For example, Eddie Lee has shown that mobile payment systems like Google wallet do not check UIDs of credit cards which allows relaying of payment transactions using NFC smart phones [2]. Next to this, it is often not possible to define or modify low-level protocol parameters, for example, the Select Acknowledge (`SAK`) information that defines the individual card type or the Frame Waiting Time (`FWT`) in the Answer to Select (`ATS`) command in ISO/IEC 14443-4. Also especially when using the smart phone as a proxy device, it is often not possible to initiate a Waiting Time Extension (`WTX`) that is necessary to request a potentially necessary amount of additional (relay) time. Changes in the ISO/IEC are in general not possible and modifications of the standard are infeasible to implement on those platforms.

B. Relay using Custom Proxy/Mole

Custom-made proxies and/or moles have the advantage that they are fast and/or very flexible depending on wether they use dedicated controllers or not. We therefore distinguish between *analog* and *digital* custom relay devices. Analog devices make use of analog circuits not involving digital-signal conversions which makes the relay communication very fast (far below milliseconds as, for example, shown by Thevenon et al. in [21]). They are typically passive and do not modify the content of the relay messages. Digital devices, in contrast, involve an RF-to-digital conversion including a microcontroller or processor. They can be active and allow the modification of all parameters of the RFID-network stack. In particular, they allow cloning of UIDs and the modifications of all ISO/IEC commands. For example, it is possible, as highlighted by Issovits and Hutter [11], to intentionally cause an `FWT` timeout to force the PCD to send an `R(NAK)` command which allows the proxy to re-send the message (and thus gain additional time for the relay process). Another way to extend the time, as it will be shown later in this paper, is to modify individual bits of the message such that, e.g., the CRC is incorrect or by causing bit collisions. This also forces the PCD to send an `R(NAK)` command and allows re-sending of messages according to the specification in ISO/IEC 14443-4 [10]. Note that these bit modifications can be made using custom proxies to fool standard compliant PCDs and PICCs without further interventions. Furthermore, different data rates between PCD-proxy and mole-PICC can be specified by custom devices in order to gain additional time for the relay attack. Custom-made moles also allow to generate a HF field with a frequency higher than 13.56 MHz to increase the data processing speed of PICCs, which leads to a shorter relay times. Related work also show that custom devices allow to significantly increase communication ranges, e.g., Oren et al. [16] extended the communication range between PCD and proxy to 3.8 feet (1.15 m), Kirschenbaum et al. [15] presented an approach to extend the range between mole and PICC.

As disadvantage, however, custom-made devices are usually more expensive and time and experience is needed for the design and implementation of the setup. Details about costs for custom-made proxies/moles are given in Section IV-C.

C. Related Work

Kfir et al. [14] were one of the first who pointed out the vulnerability of authentication schemes using contactless smartcards regarding relay attacks in 2005. Hancke et al. [7] presented attacks against RFID systems such as eavesdropping as well as simple relay attacks (relaying the UID of a card). For relaying the data, they used a custom communication channel between proxy and mole (FSK RF link). The distance between proxy and mole can be up to 160 feet (50 meters). Issovits et al. [11] used one NFC enabled mobile phone as mole and a special programmable RFID tag as proxy in order to perform a relay attack. As relay channel they used Bluetooth. Note that in their work, only the higher-layer communication (ISO/IEC 14443-4) was relayed. The lower-layer communication has not been relayed because of the strict timing constraints set by the standard (a few microseconds instead of a few milliseconds). In order to relay also lower-layer communication, Thevenon et al. [21] presented two different setups for relay attacks with delay times lower than 2 μs. The first setup uses a coaxial cable between two antennas. A passive matching circuit is sufficient to enable relay distances of up to 66 feet (20 meters). A wireless link is used in the second setup. In order to allow a far-field communication with very low delay, the reader signal is modulated on a carrier. At the receiver side, the signal is demodulated, the demodulated signal equals the reader signal with a low delay.

A practical long-distance relay attack was presented by

TABLE I. OVERVIEW OF DIFFERENT RELAY-ATTACK SETUPS (TYPES AND CHANNELS USED).

Channel	Custom (digital) / NFC Phone	NFC Phone / NFC Phone	Custom (digital) / Custom (analog)	Custom (analog) / Custom (analog)
Bluetooth	this work, Issovits et al. [11], Silberschneider et al. [19]	this work, Francis et al. [5], [6]		
Internet		Sportiello et al. [20]		
Custom			Hancke et al. [7]	Thevenon et al. [21]
WLAN		this work, Lee et al. [2]		

Sportiello et al. [20]. They relayed the communication of an ePassport using the Internet as a relay channel. Francis et al. [6] have used NFC smart phones for proxy and mole and Bluetooth as relay channel for their attack in order to relay unencrypted data. As they have relayed a peer-to-peer communication, where both parties are active, they did not investigate connection parameters like `FWI` or `WTX`. Vulnerabilities introduced by the usage of smart phones emulating contactless smart cards (e.g., as contactless bank cards) have been discovered by Anderson [18] in 2007. Lee [2] successfully relayed the communication of a contactless credit card transaction by using two smart phones. The fact that the system does not validate the UID enables this attack. Timing constraints are not evaluated in this work, the performance of a WLAN relay channel is sufficient in order to succeed with the attack.

Table I gives an overview of related work and this work and their used setup and relay channel. We used a modified version of the digital custom-made proxy of Issovits and Hutter [11] but relayed encrypted communication and explicitly highlighted the advantages of custom proxies in this work. Silberschneider et al. [19] also applied a custom-made proxy, but did not compare it to the WLAN relay channel nor did a detailed timing analysis. Francis et al. [5], [6] used a Bluetooth relay channel between two NFC phones but they did not discuss the limitations of this setup compared to custom devices that is a main contribution of this work. Finally, Lee et al. [2] also used WLAN as relay channel between two NFC phones but we extend this setup in this work by presenting a "three-phones-in-the-middle" scenario which leads to an improved relay distance.

D. Terminology

Figure 1 shows the communication flow for a standard smart card (upper plot) and a relay-attack scenario (lower plot). Throughout the rest of the paper, we use the following terminology:

- t_{PP} represents the duration from proxy request to mole response,
- t_{RR} represents the duration from PCD/reader request to proxy response, and
- t'_{RR} denotes the time from PCD/reader request to PICC response (no relay).

The most important parameter is the t_{PP}, the time between proxy request and mole response. This time covers the entire relay chain. This value varies for different relay channel types and proxy/mole devices.

IV. ATTACK SCENARIOS AND USED SETUPS

For the relay attacks described in the following, we focus on relaying an AES authentication process between a PCD and an ISO/IEC 14443 Type A PICC. First, we describe the authentication process in a detail. Second, we describe our used setups using two and three NFC smart phones. Finally, we introduce our custom proxy device.

A. Relaying an AES Authentication Process

Authentication is required for many applications such as access control, ticketing, or e-passports which makes it an attractive target for relay attacks. Most of the commercially available smart cards that include security features, e.g., the Mifare DESFire card family, implement the authentication process according to ISO/IEC 9798-2 [9]. This standard specifies a challenge-response protocol where the PCD sends a random challenge to the PICC that performs an encryption operation on the challenge (and possibly optional data in addition). The PICC sends the answer back to the PCD which can then decide if the PICC is authentic or not by decrypting and evaluating the answer of the PICC.

Figure 2 shows the commands exchanged during our relay attacks. In a first step, the PCD starts the initialization and anticollision phase and sends a `REQA` and `SELECT` command to the proxy device. The proxy answers with its UID and changes into *ACTIVE* state. After that, ISO/IEC 14443-4 commands are exchanged using the block format of the transmission protocol. The encapsulated messages are formatted according to ISO/IEC 7816-4 APDUs [8]. The PCD sends the challenge to the proxy using an *INTERNAL_AUTHENTICATE* command. The proxy forwards the challenge to the mole over either Bluetooth or WLAN, which then forwards the challenge to the PICC using NFC. In the meanwhile, the (custom digital) proxy might send a Waiting Time Extension (WTX) in cases where the relay communication is slow in order to request an additional response time for the PICC. The answer of the PICC is then sent back to the PCD over the NFC and Bluetooth/WLAN relay link.

B. The "Phones-in-the-middle" Attack

First, we use two NFC-enabled smart phones and perform a relay attack using Bluetooth (as similarly done by Francis et al. [6]) as a reference attack. Second, we performed the same attack over a WLAN link and compare the results in terms of performance and relay range. Finally, we add another NFC phone that acts as a WLAN access point to extend the relay communication link. The same evaluation is done in this case.

The Proxy. As a proxy, we used a Google Nexus S smart phone. This smart phone runs the Android operating system and it has an NFC chip integrated which enables it to act as an NFC reader. In order to allow emulation of a PICC, we made use of a modified operating system kernel called CyanogenMod 9.1. Card emulation is per se only supported by the Android operating system Version 4.4 (*KitKat*). The Google Nexus S smart phone does not receive an OS update to

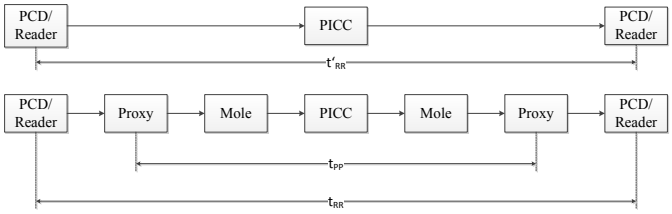

Fig. 1. The upper part shows a scenario without relaying and the lower part shows a scenario performing a relay attack.

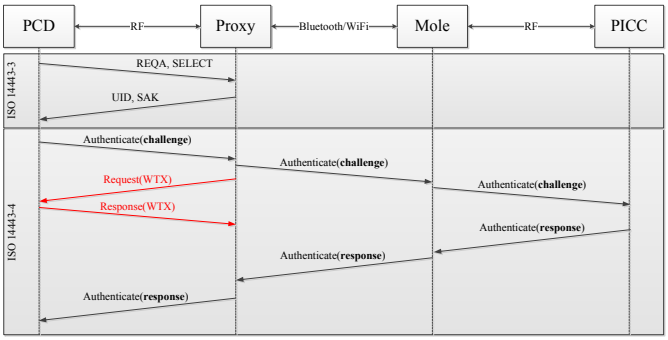

Fig. 2. Relay communication flow during an AES authentication process.

Version 4.4, so we decided to install and use CyanogenMod 9.1 for that purpose. A detailed description for that process as well as how to use the card emulation can be found in [3]. Note that the proxy device is under control of the attacker so the required modifications regarding the operating system do not influence the applicability of the attack. On the other hand, newer devices running Android 4.4 can make use of the host card emulation (HCE) feature in order to act as a proxy in relay attacks[4].

As already described in Section III, the use of NFC phones comes along with certain restrictions. One of these restrictions is that the UID of the smart phone is generated randomly for every new `SELECT` command of the PCD. In our scenario, the UID is four bytes long where the first byte is fixed to the manufactory value 0x08. We therefore assume an RFID system that does not check the UID but only verifies higher-level protocol commands. In particular, the used smart phone only allows to send ISO/IEC 14443-4 commands using I-blocks. As an additional drawback, it is not able to send Waiting Time Extensions (WTX) to the PCD since this is automatically handled by the device in hardware. Our smart phone (Google Nexus S with Cyanogenmod 9.1 OS) further sets the FWT to 77.3 ms. It showed, however, that this value is sufficient for our relay attacks as demonstrated in the next section.

The software application that runs on our proxy works as follows. As soon as the proxy is in the reading range of the PCD and after it has been successfully selected, it listens to the commands of the PCD. If an *INTERNAL_AUTHENTICATE* command is received, the challenge is forwarded to the mole. At this point the relay channel can be set to either WLAN or Bluetooth. The received response from the mole is then forwarded to the PCD using the NFC interface of the smart phone. The time required for the relay communication (t_{PP}) is measured using the time measurement ability of Java (`System.currentTimeMillis()`).

The Mole. As a mole, we also used a Google Nexus S smart phone (running Android Version 4.1.2). Note that in case of the mole, no modifications of the operating system are required because no card emulation feature is required. Instead, the mole needs to act as a conventional PCD.

The application on the mole works as follows. First, the PICC is selected and as soon as an *INTERNAL_AUTHENTICATE* command is received, the message is forwarded to the PICC via the NFC/RFID channel. After reception of the response from the PICC, the data is simply forwarded to the proxy on the selected relay channel.

Three Phones in the Middle. In this scenario, we added another NFC phone between the proxy and the mole to act as a WLAN access point (AP). For this purpose, we also used another Google Nexus S and enabled the "Mobile WLAN-Hotspot" feature of that device. If enabled, the proxy as well as the mole can connect to this AP and therefore extend the relay channel by a factor of two. Note that in this scenario, no Internet connection is required, instead, the communication is routed over the AP device under control of the attacker. This scenario increases the applicability of the relay attack due to a larger relay distance.

C. A Custom Relay Proxy

Instead of using an NFC phone as a proxy, we now use a custom-made proxy device which is similar to the OpenPICC [1] or Proxmark [12] devices. Our custom proxy as well as the Proxmark RFID simulator can be assembled for less than $400. The development of our proxy was an iterative process lasting several years, including hardware improvements (analog front-end, interfaces, etc.) and software improvements (increasing the number of supported protocols and applications).

Figure 3 shows our custom-made proxy. The main parts of the proxy are a programmable Atmel AVR microcontroller—therefore analog signals are converted to digital signals and vice versa during the communication. The left board in Figure 3 shows the microcontroller board, which implements the ISO/IEC 14443 protocol (type A) and allows sending and receiving of higher-level APDUs. The analog part consists of an HF antenna and a simple modulation/demodulation circuit that is connected to the I/O of the AVR microcontroller. The analog part is placed on the main board shown on the right side in Figure 3. Furthermore, we connected a self-made Bluetooth module to the microcontroller (using an available RS232 interface of the AVR) to allow sending and receiving of data with a mole.

Since the microcontroller is freely programmable, we are able to set custom UIDs (4, 7, or 10 bytes). Thus, we are able to completely clone the UID of an existing PICC. Furthermore, we have the possibility to make any modifications in the lower-protocol level, i.e., setting the same `SAK` or `FWT`

[4]More information about the HCE feature can be found at http://developer.android.com/about/versions/android-4.4.html.

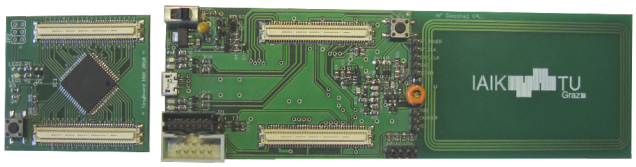

Fig. 3. The custom made proxy. Left: microcontroller board, Right: main board including analog part.

values as the victim's PICC. For example, setting the Frame Waiting Integer (FWI) to the maximum value corresponds to a FWT of nearly 5 seconds which is far enough to relay a communication around the world using the Internet as a relay channel [20], for instance. The proxy also allows sending of Waiting Time Extensions (WTX) in order to increase the FWT for the subsequent command.

Using the custom-made proxy, the UID of the PICC can be first challenged by the mole which sends the UID to the proxy. The proxy can then simply clone the UID before the actual initialization and anticollision of the PCD, in contrast to the usage of a smartphone as proxy. This cloning can be done if the UID is considered as a known constant. If it is not a known constant and if the UID will be checked by the RFID system (especially in cases where only a single pass is possible), we propose the following anticollision-time extension. The proposed extension allows to gain additional time during the anticollision of PICCs to successfully relay the (unknown) UID between PICC and proxy before answering to the PCD with the correct UID of the victim. The proposal is fully standard compliant and does not require any modifications on the PCD or PICC side.

ISO/IEC-Compliant Anticollision Time Extension. In order to increase the response time during PICC anticollision, we propose to induce single bit faults during the anticollision loop. The idea is very similar to the *blocker tag* proposal of Juels, Rivest, and Szydlo [13] that is used to successfully block individual UIDs for privacy-preserving reasons. In our case, however, we do not entirely block the cards but we rather exploit the fact that the time for the anticollision loop can be extended if collisions occur which can be accomplished using our custom-made proxy.

In the following, we briefly introduce the anticollision loop (binary search tree algorithm or often referred to as "tree walking" protocol) as specified in ISO/IEC 14443-A [10]. The idea, however, can be also applied for slotted ALOHA (type B) or other (proprietary) singulation protocols (e.g., UHF EPC Gen2). The following algorithm is a recursive depth-first search in a binary tree and works as follows.

1) First, the PCD sends a SELECT command (SEL, i.e., $0x93$ in case of a 4-byte UID) followed by the actual Number of Valid Bits (NVB)[5], which is per default $0x20$.
2) After that, all PICCs in the field of the PCD answer with their UIDs. If there are several PICCs in the field of the PCD, collisions might occur (can be recognized by incorrect field modulations). In this case, the PCD identifies the bit position of the collision and re-sends all UID bits up to the position where the collision occurred and adds a 0 or 1 (the PCD sets the NVB accordingly).
3) Now, only PICCs that start with the same UID bit prefix answer with the remaining bits. This is repeated recursively until all PICCs are identified.

[5]The Number of Valid Bits (NVB) defines the number of already correctly received UID bits of a PICC.

Let's assume a victim's PICC with a 4-byte UID. Then, there are $2^k = 2^{32}$ possible UIDs that can be identified by the given algorithm, where k denotes the binary tree depth. Now, further assume that our custom-made proxy device is able to intentionally cause bit collisions by sending a 0 or 1 simultaneously. Then, a proxy can cause bit collisions for the first x bits of all possible UIDs to gain additional relay time, where x represents the required tree-search depth to succeed a relay attack in a given time.

An example. Let's estimate the runtime needed to send one anticollision command. According to the standard, this needs $FDT_{PCD\_to\_PICC} + T_{Anticollision\_frame} + FDT_{PICC\_to\_PCD}$ microseconds. The $FDT_{PCD\_to\_PICC}$ represents the FDT between the PCD and PICC, i.e., $91.15\,\mu$s or $86.43\,\mu$s dependent on the last bit transmitted by the PCD. The $T_{Anticollision\_frame}$ represents the time needed for transmitting the anticollision frame, i.e., $9.4 \cdot (20 + NVB)\,\mu$s. Finally, $FDT_{PICC\_to\_PCD}$ represents the FDT between the PICC and PCD, i.e., $86.43\,\mu$s. So the amount of time needed to transmit and receive an anticollision command (without considering the PCD processing time to prepare the anticollision commands) is between $361\,\mu$s $(= 2 \cdot 86.43 + 9.4 \cdot (20 + 0))$ and $554\,\mu$s $(= 91.15 + 9.4 \cdot (20 + 20) + 86.43)$.

Now, if a proxy would like to extend the anticollision time to, for example, 100 milliseconds, and if we assume $450\,\mu$s for a single anticollision command, the proxy has to cause bit collisions for the first x bits of the UID, i.e.,

$$2^x \cdot 450 = 100\,000 \rightarrow x = log_2\left(\frac{100\,000}{450}\right) = 7.796. \quad (4)$$

V. Results

In this section, the results of the performed relay attacks are presented. First, we present the results obtained by using NFC smart phones. After that, results of our custom proxy are discussed.

A. "Phones-in-the-middle"

In a first experiment, we used a Bluetooth connection as a relay channel between proxy and mole. 500 authentication runs have been performed using this setup. We measured the time between receiving the request of the reader and receiving the response of the mole, i.e., t_{PP}, on our proxy device. Figure 4 shows the result of this experiment. The distance between proxy and the mole was about 7 feet (2 meters). The mean value for t_{PP} for the 500 measurements is 67 ms and the minimum value is 55 ms. 90 % of the values are in the range between 55 ms and 80 ms and 99 % of the values are in the range between 55 ms and 100 ms.

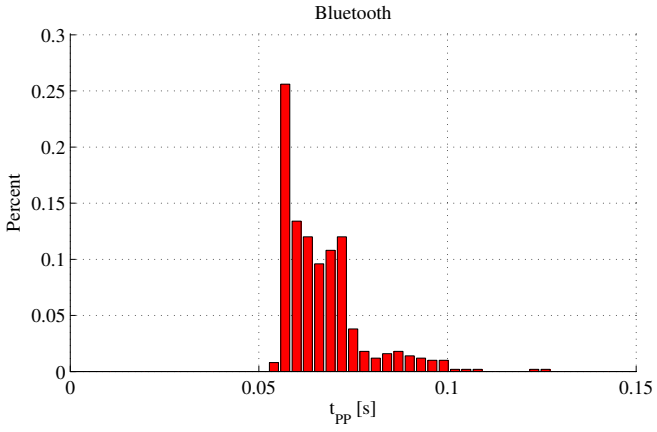

Fig. 4. Result of 500 measurements of t_{PP} with constant distance between proxy and mole using Bluetooth as relay channel.

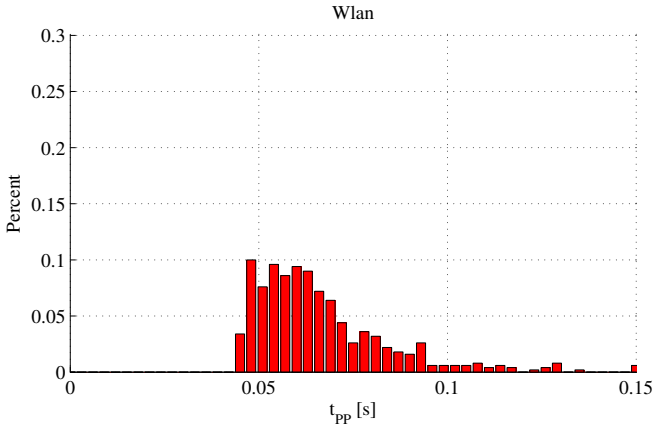

Fig. 5. Result of 500 measurements of t_{PP} with constant distance between proxy and mole using WiFi as relay channel.

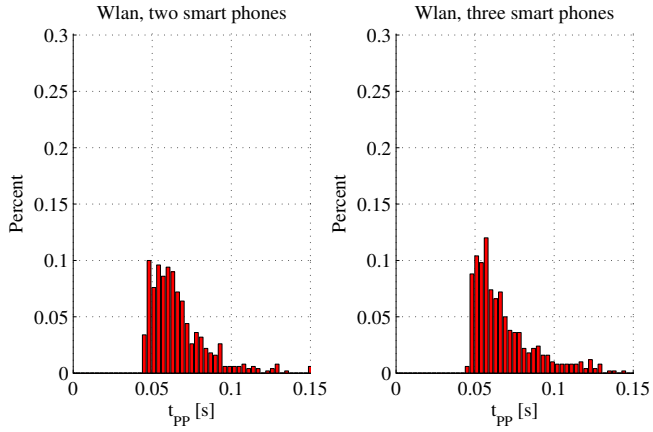

Fig. 6. Comparison of t_{PP} for the scenario with only two smart phones (left) and three smart phones (right).

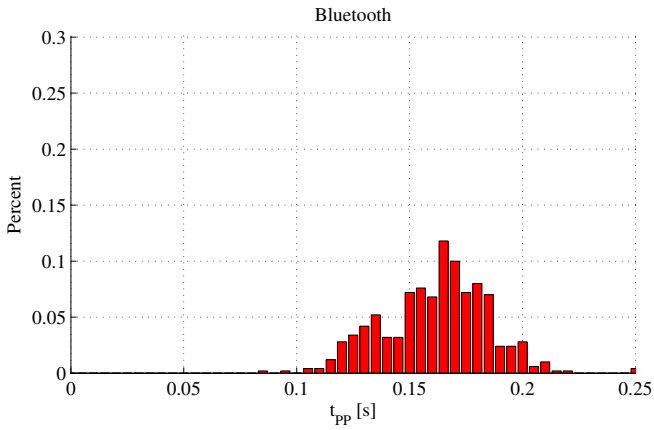

Fig. 7. Result of 500 measurements of t_{PP} with constant distance between proxy and mole using Bluetooth as relay channel and a programmable transponder as proxy.

After that, we increased the distance between proxy and mole until the connection got lost. It was possible to increase the distance to about 130 feet (40 meters) without loosing the Bluetooth connection for our setup. It showed that the distance between proxy and mole does not significantly has an influence on t_{PP}.

In a second experiment, we relayed the communication over WLAN instead of Bluetooth. We enabled WLAN on both proxy and mole, at which one device acts as a WLAN access point and the other one connects to it. Again t_{PP} was measured for 500 authentication runs while the distance between proxy and mole was set to 7 feet (2 meters). The result of this experiment is shown in Figure 5. The mean value for t_{PP} for the 500 measurements is 67 ms and the minimum value is 46 ms. 90 % of the values are in the range between 46 ms and 80 ms and 99 % of the values are in the range between 46 ms and 129 ms.

After this experiment, we also increased the distance between proxy and mole to make a comparison to Bluetooth. Using WLAN, a relay distance of up to 197 feet (60 meters) was possible. The distance does not significantly influence t_{PP}.

"Three Phones-in-the-middle". We placed a third NFC phone between proxy and mole and enabled a WLAN access point. It showed that this additional phone theoretically doubles the relay distance and does not significantly increase t_{PP}. With this setup, we achieved a maximum relay distance of over 360 feet (110 meters). A comparison between two and three "phones-in-the-middle" is shown in Figure 6. The mean value for t_{PP} for the 500 measurements using three smart phones is 68 ms and the minimum value is 46 ms. 90 % of the values are in the range between 46 ms and 90 ms and 99 % of the values are in the range between 46 ms and 130 ms.

B. Bluetooth vs. WLAN

With WLAN we were able to achieve lower t_{PP} values compared to Bluetooth (46 ms vs. 55 ms). When keeping in mind the maximum delay time of about 5 seconds allowed by the ISO/IEC standard, the difference of 9 ms between the two technologies does not affect the application of the relay attack. The fact that the FDT value cannot be influenced and is fixed to 77.3 ms for the card emulation on the smart phone shows that in this scenario a fast connection is still advantageous. With the WLAN connection, 76 % of the t_{PP} values were below FDT and for the Bluetooth connection 88.4 % of the t_{PP} values were below FDT. The variation of the t_{PP} values for subsequent authentication runs is higher in the WLAN scenario but even in the worst case t_{PP} did not exceed 130 ms. With WLAN and

TABLE II. COMPARISON OF RESULTS BETWEEN BLUETOOTH AND WLAN USING NFC PHONES.

		Bluetooth	WLAN, 2 phones	WLAN, 3 phones
$t_{\text{PP min}}$	[ms]	55	46	46
$t_{\text{PP mean}}$	[ms]	67	67	68
Interval 90 %	[ms]	55 - 80	46 - 80	46 - 90
Interval 99 %	[ms]	55 - 100	46 - 129	46 - 130
Max. distance	[m]	40	60	110

TABLE III. RESULTS FOR OUR CUSTOM-MADE PROXY USING A BLUETOOTH CHANNEL.

		Bluetooth
$t_{\text{PP min}}$	[ms]	86
$t_{\text{PP mean}}$	[ms]	162
Interval 90 %	[ms]	86 - 187
Interval 99 %	[ms]	86 - 212
Max. distance	[m]	40

two smart phones, distances of up to 197 feet (60 meters) were achieved in our experiments which is a bit more compared to Bluetooth. By introducing a third smart phone acting as an access point, the distance could nearly be doubled to 360 feet (110 meters). In Table II, the achieved results are summed up.

C. Custom Relay Proxy

We evaluated the performance of our custom proxy by measuring the relay timings for 500 authentication runs. The result is depicted in Figure 7. The distance between proxy and mole was set to 7 feet (2 meters) as in the previous experiments and we did not change the distance during this experiment. The mean value for t_{PP} is 162 ms and the minimum value is 86 ms. 90 % of the values are in the range between 86 ms and 187 ms and 99 % of the values are in the range between 86 ms and 212 ms. In order to find the maximum distance for relaying the communication, the distance between proxy and mole has been increased step by step. The mole lost the Bluetooth connection to the proxy at a distance of approximately 132 feet (40 meters) in our experiment. Table III shows the exact relay timings.

VI. CONCLUSION

In this paper, we pointed out how practical relay attacks can be improved when using a custom-made proxy compared to NFC-enabled smart phones. We started with a discussion of the limitations which arise when using a smart phone as a proxy device. Some of these limitations are: the UID cannot be set to a fixed, predefined value; adaption of low-level ISO/IEC protocol parameters is not possible; no active/direct request for Waiting Time Extensions; no way to modify lower-level RFID protocol commands. We emphasized that these limitations can be circumvented by using custom-made proxies and presented practical results of attacks using a microcontroller-based (low cost) device.

The results show that practical relay attacks performed with two NFC smart phones pose a real threat due to the fact that the introduced delay time is below the maximum, tolerated delay time for both evaluated relay channels (Bluetooth, WLAN). Furthermore, the "three-phones-in-the-middle" approach allows to nearly double the relay distance without the need for a public network. Delay times are not (noticeable) affected by this modification. Our custom-made proxy is highly flexible and allows more sophisticated attacks than using NFC-enabled smart phones.

REFERENCES

[1] Bitmanufactur. OpenPICC. http://www.openpcd.org, 2013.

[2] Eddie Lee. NFC Proxy. http://sourceforge.net/p/nfcproxy/wiki/Home/, 2012.

[3] N. Elenkov. Emulating a PKI Smart Card with CyanogenMod 9.1. http://nelenkov.blogspot.it/2012/10/emulating-pki-smart-card-with-cm91.html, October 2012.

[4] M. Feldhofer, S. Dominikus, and J. Wolkerstorfer. Strong Authentication for RFID Systems using the AES Algorithm. In M. Joye and J.-J. Quisquater, editors, *CHES, Cambridge, MA, USA, August 11-13*, volume 3156 of *LNCS*, pages 357–370. Springer, Heidelberg, August 2004.

[5] L. Francis, G. Hancke, and K. Mayes. A Practical Generic Relay Attack on Contactless Transactions by Using NFC Mobile Phones. *International Journal of RFID Security and Cryptography (IJRFIDSC)*, 2:92–106, 2013.

[6] L. Francis, G. Hancke, K. Mayes, and K. Markantonakis. Practical NFC Peer-to-peer Relay Attack using Mobile Phones. In *Radio Frequency Identification: Security and Privacy Issues*, pages 35–49. Springer, Heidelberg, 2010.

[7] G. P. Hancke. Practical Attacks on Proximity Identification Systems. In *IEEE Symposium on Security and Privacy – S&P, Berkeley/Oakland, California, USA, May 21-24*, pages 328–333. IEEE Computer Society, May 2006.

[8] International Organisation for Standardization (ISO). ISO/IEC 7816-4: Information technology - Identification cards - Integrated circuit(s) cards with contacts - Part 4: Interindustry commands for interchange. http://www.iso.org, 1995.

[9] International Organisation for Standardization (ISO). ISO/IEC 9798-2: Information technology – Security techniques – Entity authentication – Mechanisms using symmetric encipherment algorithms, 1999.

[10] International Organization for Standardization (ISO). ISO/IEC 14443: Identification Cards - Contactless Integrated Circuit(s) Cards - Proximity Cards, 2009.

[11] W. Issovits and M. Hutter. Weaknesses of the ISO/IEC 14443 Protocol Regarding Relay Attacks. In A. Collado and M. Bozzi, editors, *Conference on RFID-Technologies and Applications – RFID-TA, IEEE International Conference, Barcelona, Spain, September 15-16*, pages 335–342. IEEE, September 2011.

[12] Jonathan Westhues. Proxmark.org A Radio Frequency IDentification Tool. http://www.proxmark.org/, 2013.

[13] A. Juels, R. L. Rivest, and M. Szydlo. The Blocker Tag: Selective Blocking of RFID Tags for Consumer Privacy. In *10th ACM Conference on Computer and Communication Security, Washington, DC, USA, October 27-30*, pages 103–111. ACM Press, October 2003.

[14] Z. Kfir and A. Wool. Picking Virtual Pockets using Relay Attacks on Contactless Smartcard Systems. In *First International Conference on Security and Privacy for Emerging Areas in Communications Networks (SecureComm 2005), Athens, Greece, 5-9 September 2005, Proceedings*, pages 47–58. IEEE Computer Society, September 2005.

[15] I. Kirschenbaum and A. Wool. How to Build a Low-Cost, Extended-Range RFID Skimmer. Cryptology ePrint Archive (http://eprint.iacr.org/2006/054.pdf), Report 2006/054, 2006.

[16] Y. Oren, D. Schirman, and A. Wool. Range Extension Attacks on Contactless Smart Cards. In *Computer Security–ESORICS 2013*, pages 646–663. Springer, 2013.

[17] S. Research. NFC World. http://www.nfcworld.com.

[18] Ross Anderson. RFID and the Middleman. In *Financial Cryptography and Data Security*, pages 46–49. Springer, Heidelberg, 2007.

[19] R. Silberschneider, T. Korak, and M. Hutter. Access Without Permission: A Practical RFID Relay Attack. In *Austrochip, 21st Austrian Workshop on Microelectronics, Linz, Austria, October 10*, pages 59–64. Johannes Kepler University of Linz, 2013.

[20] L. Sportiello and A. Ciardulli. Long Distance Relay Attacks. In *Workshop on RFID Security – RFIDsec, Graz, Austria, July 9-11*, 2013.

[21] P.-H. Thevenon, O. Savry, and S. Tedjini. On the Weakness of Contactless Systems under Relay Attacks. In *19th International Conference on Software, Telecommunications and Computer Networks – SoftCOM*, pages 1–5, 2011.