Confidence in Smart Token Proximity: Relay Attacks Revisited

G.P. Hancke[a], K.E. Mayes[a], K. Markantonakis[a]

[a]*ISG Smart Card Centre*
Royal Holloway, University of London
Egham TW20 0EX, UK

Abstract

Contactless and contact smart card systems use the physical constraints of the communication channel to implicitly prove the proximity of a token. These systems, however, are potentially vulnerable to an attack where the attacker relays communication between the reader and a token. Relay attacks are not new but are often not considered a major threat, like eavesdropping or skimming attacks, even though they arguably pose an equivalent security risk. In this paper we discuss the feasibility of implementing passive and active relay attacks against smart tokens and the possible security implications if an attacker succeeds. Finally, we evaluate the effectiveness of time-out constraints, distance bounding and the use of a additional verification techniques for making systems relay-resistant and explain the challenges still facing these mechanisms.

Key words:
Relay attack, RFID, smart card, token, contactless, near-field communication

1. Introduction

Smart tokens are often used for the purpose of proximity identification in secure systems. Contactless tokens use near-field communication, usually limiting the operating range of most readers to less than 10 cm and contact tokens, as the name suggests, have to physically touch the reader. If

Email address: Gerhard.Hancke@rhul.ac.uk (G.P. Hancke)

the reader successfully communicates with a token it is therefore assumed that the token is close physical proximity. Using only these physical characteristic of the communication channel, however, is not suitable for securely proving the proximity of a token. An attacker can use a proxy-token and proxy-reader to relay the communication between a legitimate reader and token over a greater distance than intended, thereby tricking the reader into believing that the real token is in close proximity. A successful relay attack therefore allows an attacker to temporarily possess a 'virtual clone' of a token, thereby allowing him to gain the associated benefits.

Relay attacks are not a new concept. The attack scenario was already proposed in the 1970s [1] and more recently relay attacks, known in this context as 'wormhole' attacks, have also become a recognised threat in wireless network security [2]. The possibility of relay attacks have been discussed briefly in security surveys and threat frameworks concerning Radio Frequency IDentification (RFID), e.g. [3, 4], but at the same time there are several examples, including comprehensive industrial and government guidelines, of survey publications that do not take relay attacks into account, e.g. [5, 6, 7]. The reason for failing to consider relaying as a threat could be that security experts tend to treat this attack as a mixture of conventional man-in-the-middle and skimming attacks, which can be prevented with application layer authentication or physical security mechanisms. Relay attacks, however, are not that easy to defend against and even though physical mechanisms, such as the shielding of contactless tokens, could prevent certain attack scenarios, any application layer security is effectively circumvented. As a result, it is irrelevant whether the system implements secure authentication and encryption mechanisms. To our knowledge the only cryptographic mechanism considered to be suitable for preventing relay attacks are distance-bounding protocols, which will detect the additional delay introduced by the attacker. These protocols are, however, not implemented in current systems and existing time-out constraints imposed on the communication between a reader and token has been shown to be ineffective in detecting this extra attack delay. The attack could be made more difficult by implementing 'two factor' authentication, although this method cannot prevent the attack entirely and is considered impractical in certain systems.

This paper serves as an overview of the recent academic work related to relay attacks, detailing both our own work and that of other researchers,

which highlight the practical feasibility and security implications of these attacks. Our primary motivations are to emphasise the challenges this attack poses to proximity identification and to assist system engineers to fully understand the risks and attack techniques when selecting or upgrading smart card technologies. In Section 2 we discuss active and passive relay attacks, first elaborating on the logical attacks before examining the feasibility of practically implementing these against current systems in Section 3. Section 4 considers the security implications and presents possible exploitation scenarios. Finally, in Section 5 we describe some of the countermeasures proposed and comment on their effectiveness.

2. Relay attacks

In 1976, Conway [1] first proposed the Grand Master Chess problem, which described how a person who does not know the rules of chess could play against two grand masters by challenging both of them to a postal game. The player would then simply forward the move received from one grand master to the other, effectively making them play against one another. A relay attack, or 'mafia fraud' as it was first referred to by Desmedt *et al.* [8], is an extension of this scenario to security protocols. For example, an attacker can circumvent an authentication protocol by simply relaying a challenge to a legitimate prover, who will provide him with the correct response, which can then be relayed back to the verifier. It does not matter what application layer protocols or security algorithms are used as the attacker just relays all the application layer data, thereby ensuring that both the verifier and the prover always receive the data they expect.

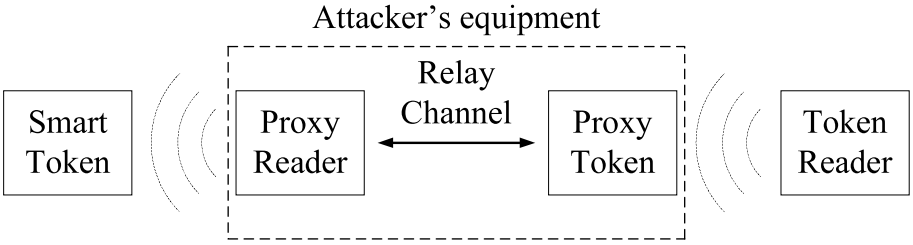


Figure 1: Basic relay attack setup

To execute a relay attack the attacker needs two devices, which act as a token and a reader respectively. These devices are connected via a suitable

communication channel in order to relay information over a greater distance. For example, the basic relay setup for attacking a contactless system is shown in Figure 1. The proxy-reader is used to communicate with the real token, while the proxy-token is placed near the real reader. Any information transmitted by the reader is received by the proxy-token and relayed to the proxy-reader, which will transmit the information to the token. The token assumes that it is communicating with the reader and responds accordingly. The token's response is then relayed back to the proxy-token, which will transmit the information to the reader. The intention of the attacker is to ensure that the reader is unable to distinguish between the real token and the proxy. If he succeeds the reader will assume that the token and its associated owner are in close proximity and grant access to the attacker.

A simple attack against a system implementing application layer cryptographic mechanisms could be illustrated using the example of a a door access control system using smart tokens. The door reader and the token share a secret key K and the reader authenticate the token presented before unlocking the door. An attacker sets up a relay attack and approaches the door, where the reader challenges him by asking his 'token' to encrypt a challenge C. The attacker forwards C to an accomplice with access to a legitimate token, for example a token in the pocket of an employee having his lunch in a nearby cafeteria. The accomplice challenges the legitimate token and learns the correct response $E_K\{C\}$, which be sends back to the attacker standing at the door. As the attacker's 'token' responds with the correct answer the reader assumes that the legitimate token is present and unlocks the door for the attacker to gain access. If the reader wanted to read some further access conditions off the token, e.g. $E_K\{\text{Read Sector AC}\}$, the attacker and his accomplice relays information in the same way and the attacker's 'token' would be able to respond with the expected $E_K\{\text{AC's content}\}$. The attacker never needs to know the plaintext data or the key K as long as he and his accomplice can continue relaying the respective messages between the reader and the legitimate token. It does therefore not matter if the data is encrypted using the Advanced Encryption Standard AES with a 256-bit key, or a weak proprietary cipher with a 32-bit key as the resultant ciphertext of either can be relayed just as easily. The success of the attacker is therefore independent of the application layer protocol and encryption algorithm used and as a result application layer cryptographic mechanisms are ineffective at preventing relay attacks.

Relay attacks can be classified as either passive or active. When executing a *passive* relay attack the attacker does not modify the data in transit. To be successful, the attacker only needs to relay the communication between the token and the reader for the duration of the transaction. The passive attack is limited when compared to conventional man-in-the-middle attacks, since the attacker cannot modify or access any data he relays unless further flaws exist in the protocols or algorithms used. Despite this limitation, relay attacks still pose a threat to systems that provide privileges and services if it simply manages to perform a successful authentication sequence with the token presented. For example, an access control system will open a door and a vending machine will dispense an item if presented with what it perceives to be a valid token in close proximity. A relay attack setup is, however, also an ideal platform for executing a 'real-time' man-in-the-middle attack. During this *active* relay attack the adversary could also exploit an existing weakness in the security mechanisms of the system to modify the transaction data. Memory and logic tokens recommended for high-volume closed payment and access control systems are probably more vulnerable than high security micro-controller tokens, as they implement limited security mechanisms due to resource and cost constraints. For example, if a token only implements an authentication function but the subsequent data is not encrypted the attacker could relay the authentication exchange and then modify the data that follows.

3. Practical Implementation

The theoretical principles of a relay attack is quite simple but the attacker must still overcome timing restrictions and the practical engineering challenges of relaying signals between the participants. Relay attacks, however, are practically feasible against contactless and contact tokens as example demonstrations in 2005 [10] and 2007 [9] showed. There are several factors that aid relay attacks in the contactless environment. One of the perceived benefits of contactless technology is convenience for the customer. Contactless tokens do not have to be oriented correctly and inserted into readers like contact cards and the time it takes the customer to perform an action is therefore reduced. To preserve this time advantage contactless systems rarely require further user interaction, such as entering a Personal

Identification Number (PIN), when processing a transaction. It is also a possibility that an attacker could also activate a contactless token while still in the victim's pocket, wallet or bag by using a modified reader with a larger operating range [21]. The attacker does, therefore, not need to convince the victim to hand over his token for a period of time, or to insert it into a proxy-reader and enter a PIN, as would be the likely case for a relay attack against contact tokens. The contactless operation also makes the construction of the proxy-token easier. People often scan their wallet, purse or bag containing the token, which means that an attacker never needs to reveal his hardware. In an attack on a contact token the attacker has to take out his proxy-token in order to insert it into a reader, for example at a point-of-sale terminal in the presence of a vendor, so it has to closely resemble a real card. We describe our implementation of a contactless relay attack in Section 3.1 followed by a brief overview of the attack against contact cards described in [9] in Section 3.2.

3.1. Contactless Systems

An attacker can choose a number of different approaches to implement an attack depending on his skill and resources. The attacker can implement his own custom hardware for the contactless proxy token and reader or alternatively use existing hardware. Hancke [11] and Kasper [12] have both described hardware designs capable of performing a relay attack against ISO 14443A [13] systems. Alternatively, an attacker could also implement an open source reader and token design such as the OpenPCD and OpenPICC [14] and modify the hardware and software as required. If an attacker does not have the engineering skills to build hardware he could also use existing Near-Field Communication (NFC) devices when these become available. The ISO 18092, or the Near Field Communication – Interface and Protocol (NFCIP), standard [15] allows for active devices, such as cellphones, to communicate with ISO 14443 contactless devices. Such a NFC device can act as either a contactless reader or a token and should already contain additional communication channels suitable for relaying information, e.g. Wi-Fi. Even though the deployment of NFC devices is currently limited, they could provide an attacker with an ideal hardware platform for executing his relay attack. A possible relay attack setup using modified NFC devices has been proposed by Kfir *et al.* [16], although the attack was not practically implement and demonstrated. Most of the existing relay attack implementations focus on ISO 14443A systems as this is the predominant standard used in most secu-

rity sensitive applications, such as e-passports and credit cards. The relay attack scenarios does, however, apply to all tokens and it is conceivable that attackers in the future could develop attack hardware for other RFID standards, such as ISO 15693 [17], ISO 18000 [18] and Electronic Product Code Class-1 Generation 2 [19].

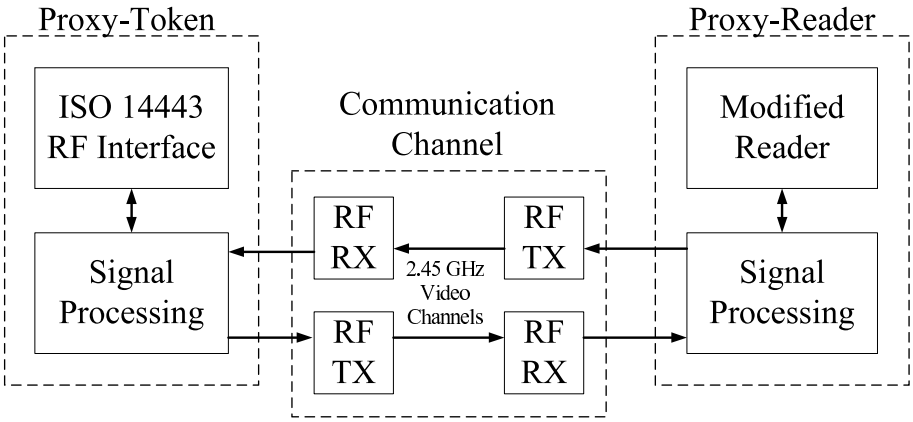
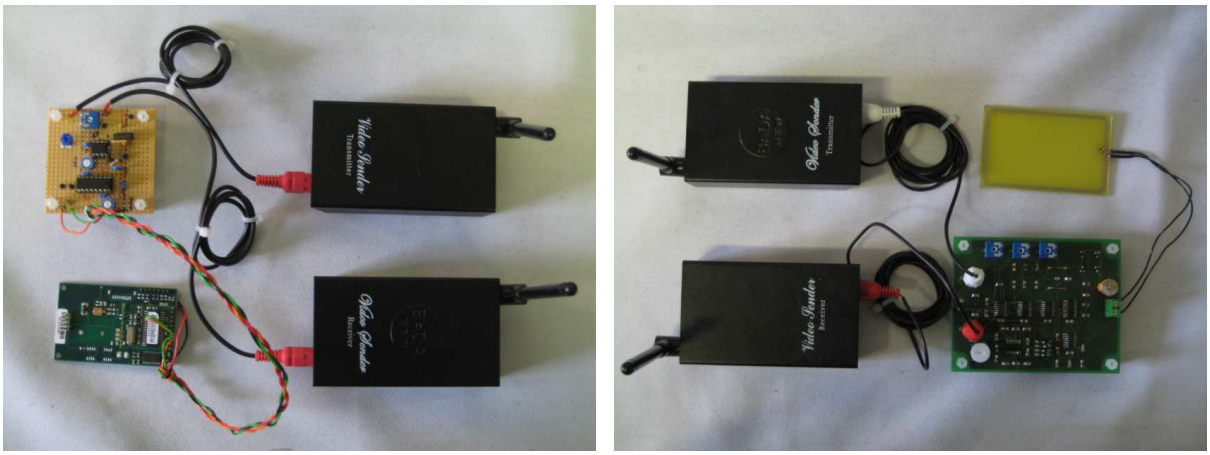


Figure 2: Example of attack implementation against a contactless system

For our relay attack investigation, we successfully reimplemented a proxy-token and a proxy-reader suitable for executing a relay attack against ISO 14443A tokens, as shown in Figure 2, based on concepts introduced in [11, 12]. Our main design consideration was to minimise the delay the hardware introduced when relaying the communication. The majority of the attack hardware was implemented with off-the-shelf components and publicly available reference designs, which were slightly modified. The necessary hardware parts were easily obtainable and the cost of the whole system was under $230, with main costs being an RFID reader for the proxy-reader and RF links for the relay channel. The proxy-token and proxy-reader hardware is shown in Figure 3.

Proxy-Reader: The main component of the proxy-reader is an RFID reader we purchased for approximately $70. The reader is build around a contactless reader integrated circuit (IC) connected to an 8-bit micro-controller. It is therefore an ideal development platform since all radio frequency (RF) and communication components are already implemented while the functioning of the reader can easily be reconfigured by simply reprogramming the micro-



(a) Proxy-Reader (b) Proxy-Token

Figure 3: Hardware for contactless relay attack

controller. To implement the proxy-reader we configured the reader to act only as a high-frequency radio front-end. The reader IC was placed in an operating mode where data supplied on a specified input pin would be modulated onto the 13.56 MHz carrier. In this configuration, the IC modulates the signal on a specified pin onto the carrier using 100% amplitude modulation. The Modified Miller encoded data, specified for reader-to-token communication in ISO 14443A, can therefore be modulated onto the carrier by connecting the input data stream to this pin. In this mode the IC also demodulates the response from the token and outputs the recovered Manchester encoded data, used for token-to-reader communication in ISO 14443A, on a specified pin. In order to interface to the relay communication channels the proxy-reader also performs some basic signal processing of the Modified Miller and the Manchester encoded data. These processing operations are described in more detail later when discussing the communication channels.

The proxy-reader's operating range is determined by the distance over which it can power the token, and its ability to receive the token's answer. This range is dependent on the transmitted power in addition to the diameter of the antenna used [20]. Although our demonstration simply used the standard operating range of our reader, the attacker would ideally want to extend the operating range to avoid detection. The attacker might therefore implement an extended range skimming attack, as already described in by

Kirschenbaum *et al.* [21], in addition to a relay attack. In this case, the RF interface would need to be modified to achieve the required operating range. The only part of the unit that needs to be covert is the antenna as it needs to be close to the victim for a short period without being noticed. The creativeness of the implementation is left to the attacker, but an antenna could be built into a briefcase, clothes, etc.

Proxy-Token: The proxy-token is basically an ISO 14443 high frequency (HF) interface, which demodulates the received communication and load-modulates the desired response onto the reader's carrier. The interface, which is partly based on a circuit design described in [22, pp 276–278], is implemented using only discrete components and basic logic ICs. For example, the sub-carrier required for load-modulating the response is generated with a binary counter and the readers communication is recovered using a simple envelope detector and threshold comparator. The proxy-token also incorporates an adjustable delay to align the response received from the proxy-reader with the start of a bit period defined by the real reader. The significance of the expected bit start times and the response timing is discussed in Section 5.1. In order to interface to the relay communication channels the proxy-reader also performs some basic signal processing of the Modified Miller and the Manchester encoded data. These processing operations are described in more detail later when discussing the communication channels.

The cost of the proxy-token is dependent on the actual way it is implemented. An attacker could implement the design on a printed circuit board, for approximately $70, or choose a simpler and cheaper method such as prototype strip board. The attacker does not need to extend the token's operating range as the device can be held in close proximity to the reader. The attacker also does not need to have a proxy-token resembling the real token as it is acceptable in contactless transactions to hold a wallet or a bag against the reader. Unlike real tokens, the proxy-token does not need to be powered by the reader and can have its own power supply thereby further simplifying the design.

Relay Channel: The communication channel relays the data using short range RF communication. For this purpose we purchased two wireless audio visual (AV) channels from eBay with an advertised range of approximately 30–100 m at a cost of approximately $35 each. One channel was used to

send data from the proxy-token to the proxy-reader and the second channel was used to send data from the proxy-reader to the proxy-token. The communication channel potentially causes the largest time delay in the system. The advantage of the AV channels were that they could transmit the Modified Miller and Manchester encoded data as received from the proxy devices, thereby eliminating the extra processing requirements and delay needed to decode, buffer and retransmit the data. The AV channels were also more reliable and easier to use than the RF components originally described in [10]. Only basic signal processing functions were required to interface the channels to the communication channels and all functions were implemented using only discrete components. The input signal had to be scaled down and the output signal passed through a comparator to correct the signal shape and level-shift the data signal to the voltage level required by the proxy devices' logic.

3.2. Contact Systems

Drimer and Murdoch demonstrated a relay attack on an EMV [23] contact payment system in the United Kingdom called "Chip and PIN" [9]. The card holder is presented with a fake terminal for PIN entry that is connected to the attacker's laptop. This laptop then communicates wirelessly with another laptop inside a backpack of an accomplice in a shop elsewhere, which is connected to a fake card that is inserted into a real point-of-sale (POS) terminal. Once both real and fake cards are inserted, the transaction data can be relayed between the real terminal and card, and authorized by the cardholder through entry of the correct PIN. Since the card holder has no feedback about the transaction, apart from information displayed on the proxy-reader, it could be possible for the attacker to authorise a transaction of any amount. For example, the attacker could be purchasing a $200 product while the cardholder thinks that he is authorising payment for a $2 cup a coffee, purchased from the attacker's accomplice. To be successful the attackers require a counterfeit terminal (proxy-reader), counterfeit card (proxy-token) and a suitable relay channel. To make our paper self-contained we briefly discuss the practical implementation of these three components, as shown in Figure 4, although we recommend that the reader refers to the original publication for a more detailed explanation.

Counterfeit card: The authors modified a genuine Chip & PIN card by connecting wires to the reverse side of the card's contact pads. The counter-

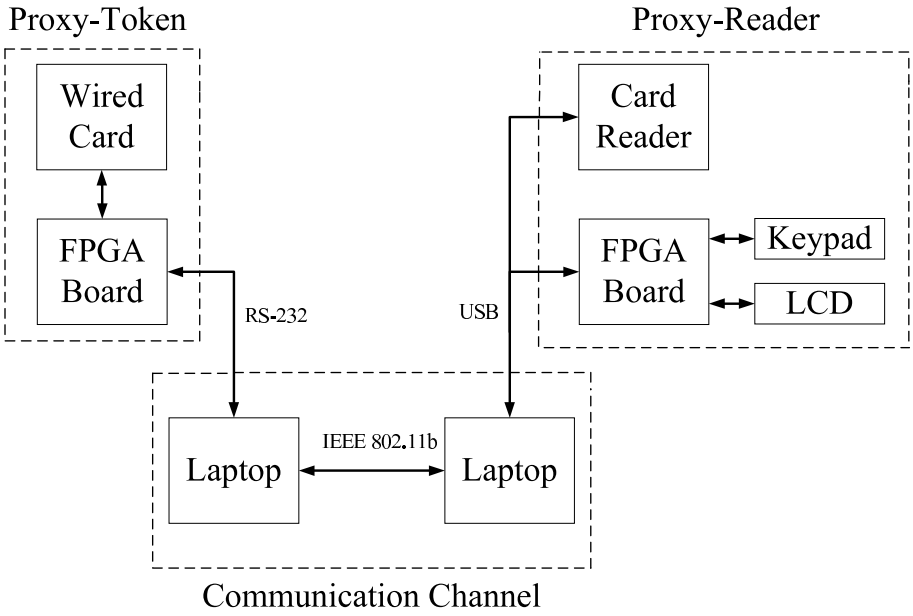


Figure 4: Example of attack implementation against a contact system as described in [9]

feit card was then connected to an FPGA development board that buffered the transaction data and translated the data between the ISO 7816 and RS-232 protocols. The total cost of the card was approximately $160.

Counterfeit terminal: The authors purchased a 'Chip & Pin' payment terminal from eBay with enough internal space to fit the hardware required for the attack. All the original internal hardware, except the keypad and liquid crystal display (LCD) screen were removed, and replaced with a smart card reader, which was connected to the existing card slot, and a small form-factor field programmable gate array (FPGA) development board. The modified terminal could record keypad strokes, display content on the LCD screen and interact with an inserted contact card, and as a result it appeared to behave like a real terminal. The total cost of the terminal was approximately $300.

Relay Channel: The counterfeit terminal and card were controlled by separate laptops via USB and RS-232 interfaces respectively, using custom control software. The laptops communicated via Transmission control Protocol (TCP) over 802.11b wireless, although the authors state that in principle this could be Global System for Mobile (GSM) or another wireless protocol.

One of the complications of this attack is that it requires careful coordination by the fraudsters in order to make sure that both cards are inserted into the respective terminals at about the same time. The attacker must have access to the token for the full duration of his interaction with the reader. Some additional synchronization is therefore needed between the attackers to present the proxy-token to a reader at the time when a suitable token has been inserted into the proxy-reader. With contact cards, there is also a chance that the merchant may notice that something is not right with the card, or even be required to handle it. The contact card attack does, however, demonstrate that current systems, even those used in security sensitive environments such as banking, do not have the necessary mechanisms in place to prevent relay attacks.

4. Security Implications

There are several ways in which an attacker can benefit from a relay attack. It should be noted that this section presents simple, worst-case scenarios to illustrate how relay attacks might be used to circumvent security measures. These scenarios could in some cases be prevented by current security mechanisms.

4.1. Passive Attack

In general, a relay attack is seen as an attack by a fraudulent third party against an honest service provide, or merchant, and a token holder. In this scenario, the attacker can masquerade as the real holder by making a proxy-token act as a virtual clone of a legitimate token. As a result, the attacker could circumvent the security of several systems, e.g. payment and access control. We have already presented on such case in Section 2 where an attacker, who wants to gain entry to a building, simply identifies an authorized token holder, possibly out to lunch, and activates his token while another attacker opens the required door. The relay attack might, however, also be used in attacks that do not involve a third party attacker.

A relay attack can also be utilized by a fraudulent merchant. A fraudulent merchant can set up the proxy-token at the reader supplied by the acquirer (likely to be his bank). His accomplice then wanders around outside with a proxy-reader and conducts payment transactions with the tokens of

unwitting victims. This attack could go unnoticed if the merchant conducts transactions, of small value, with several victims. The victims are unlikely to notice a single fraudulent transaction once they check their statements, since a sandwich or newspaper purchased from a specific merchant is not always easy to remember. Similarly, the token issuer or system operator cannot easily distinguish this attack from the regular activity on the merchant's account. However, these activities can be traced back to the merchant if the operator, bank or customer does happen to identify fraudulent transactions and sufficient punitive measures, such as the loss of their ability to accept card payments, might discourage merchants to execute such attacks. The merchant can also have several proxy-readers sending information to a single proxy-token. This allows the merchant to have multiple 'readers' without purchasing additional hardware from the acquirer, possibly circumventing expensive licensing agreements.

With token resources, like processing ability and memory size, increasing multi-application tokens are becoming more popular. A token, for example, might be required to act as both a credit/debit and a transport card with readers located in stores, or at underground rail stations. In this system, a fraudulent merchant could possibly set up a fake top-up reader to act as a proxy-reader, which then selectively relays communication to the transport authority and the debit card readers. Alternatively, it might be possible to covertly attach a small loop antenna onto the transport authority's reader, which acts as the antenna of a contactless proxy-reader relaying information to the debit card reader. A person wishing to top-up his travel purse first enters the amount he wishes to add. After payment he then presents his card to the reader for the credit to be loaded. Using the relay setup it might be possible for the merchant to also charge the debit card during the time that the card is touched to the reader. The holder is unlikely to notice the extra time taken for the debit card transaction, since both transactions can be conducted before he takes the card away. This attack could possibly be detected if the travel and payment systems look for simultaneous transactions but that would require collaboration between the two operators. In this example both applications are related to payment schemes, which are generally thought to be monitored for fraud by operators who might also make some effort to ensure that their equipment is tamper resistant, maintained and installed correctly. However, multi-application tokens potentially contain applications that require a varying degree of security. In addition, a merchant does not

necessarily have a incentive, or the skill, to regularly examine or maintain their equipment, e.g. while a customer scans his employee ID as part of a simple loyalty scheme at the coffee house next to his place of work attackers are gaining access to his office/building. In some case an extra vigilant customer could become suspicious but, as is noted in [9], it is unreasonable to expect that the token holder recognises illegitimate equipment, especially when there are so many different readers in use. For example, in June 2008 there was 157 VISA-approved point-of-sales terminals offered by 41 different vendors [24].

A fraudulent holder can also benefit from a relay attack by setting up the attack using a proxy-reader close to his own token. He then creates several proxy-tokens that all communicate with the proxy-reader. Each of the proxy-tokens now acts as a virtual clone of the original. Theoretically, this allows several 'holders' to share the same valuable token. For example, if one owner is issued with a yearly public transport pass he can issue proxy-tokens to some of his friends. Everyone can then use the same transport token, assuming that they do not travel in such a way that will alert back-end fraud detection measures to block the token. Another advantage the owner can gain by implementing an 'attack' against his own token is the ability to control the communication. The owner can therefore implement an active relay attack and selectively modify the communication. This can possible allow the attacker to exploit further vulnerabilities in the security protocols of the smart token system.

4.2. Active Attack

A number of systems still use older, or legacy, smart token technology. Some of these tokens implement proprietary cryptographic algorithms to provide security services such as authentication and encryption. In certain cases, integrity checking is also implemented using basic error detection mechanisms such as parity checking and Cyclic Redundancy Codes (CRC). Such systems could be vulnerable to an active relay attack because of weaknesses in the security mechanisms.

For example, if the data is encrypted using a stream cipher, which combines the cipher stream and plaintext in a linear way, inverting a ciphertext bit leads to the corresponding plaintext bit also being inverted. An attacker can therefore change bits in the plaintext by changing the corresponding bits

in the ciphertext. Extra cryptographic mechanisms must therefore be implemented to prevent data tampering as measures designed primarily to detect bit errors are not sufficient, i.e. CRC and parity bits can simply be modified in the same way to match the new data. This is not a new attack and has already been used against other protocols, such as Wireless Equivalent Protocol (WEP) [25].

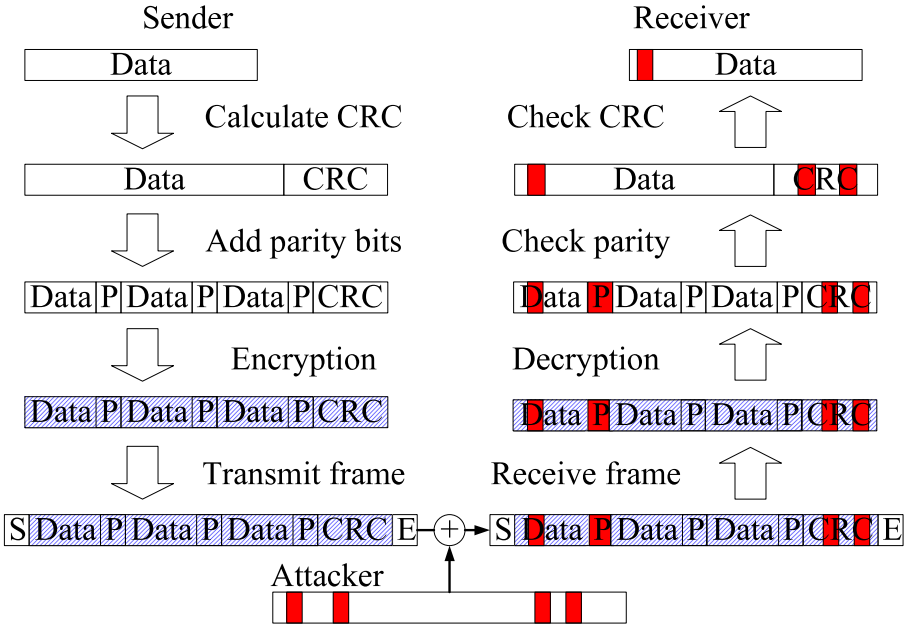


Figure 5: Modifying data bits between the token and reader

To execute this attack an attacker would require some knowledge about the communication sequence, individual frame formats, the parity bit and the CRC polynomial. Contactless system communication tend to repeat the same communication sequence during each transaction, e.g. for a travel scheme the reader might authenticate the token, read the current balance and then subtract the fare value and repeat this process each time a token is presented. As a result of these predictable exchanges, a system that does not ensure that every transaction with the same token yields different ciphertext for all messages exchanged could therefore fall prey to selective replay attacks. In the travel scheme example the attacker could just relay the authentication sequence and then replace the current balance response with a replayed

response from an earlier exchange, thereby making the reader believe that the token has more credit than is the case. In most cases, it is relatively easy to identify different commands even though some frames are encrypted. This is possible because most commands have a recognisable format, which can be obtained from a token's data sheet, example code provided by the token manufacturer or industry standards. This is useful for an attacker who wants to execute an active relay attack since he can try to figure out when the system is transmitting data of interest by means of simple traffic analysis. Next the attacker requires some knowledge about the plaintext format, which is usually not publicly published and probably differs from one system to the next. The attacker could, however, perform some trial-and-error testing to see whether a single modified bit has any effect on the system. In simple systems, an attacker might need to modify only one or two bits to change the date his pass expires or to charge his token with more credit than what he payed for. If an attacker already knows the full plaintext, the attack becomes even simpler. In this case he can simply XOR the known plaintext with the ciphertext to obtain the cipher stream and then 'encrypt' an entire new message using the resultant cipher stream. Figure 5 shows how the transmitted data frame is constructed and Table 1 shows an example of how the attacker could possibly change the encrypted data without being detected by a parity check.

Bits	Original Plaintext	Cipher Stream	Transmitted Ciphertext	Attack Pattern	Received Ciphertext	De-ciphered Plaintext
0	0	1	1	0	1	0
1	0	1	1	0	1	0
2	1	0	1	0	1	1
3	0	1	1	0	1	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0
6	0	1	1	0	1	0
7	0	0	0	1	1	1
P	0	1	1	1	0	1

Table 1: Modifying an encrypted value, with odd parity checking, from '4' to '132'.

After deciding which bits to invert in the plaintext, the attacker needs to calculate which integrity checking bits to change. If an even number of bits in a byte is changed the parity bit is kept the same. If an odd number of bits is changed the parity bit should also be changed. The CRC also needs to be changed to correspond to the new data. To do this the attacker creates a bit mask equal in length to the data block, containing only '0's, for which he calculates the CRC. The attacker then sets the bits corresponding to the plaintext bits that are to be inverted equal to '1' and again calculates the CRC. The two CRC values are XORed together and the result is appended to the bit mask. The attacker then flips all the bits in the relayed frame that corresponds to a '1' in the bit mask. If an attacker knows the plaintext the attack is much simpler. He recovers the relevant cipher stream, by XORing the plaintext and the ciphertext, and then creates a new message by XORing his plaintext to the cipher stream.

An example of a token that seems vulnerable to this attack is the Mifare Classic product supplied by NXP [26, 27]. This ISO 14443A compliant product was introduced in the 1990s and is still used in numerous access control and closed payment systems today [28]. It is basic memory token with a few simple commands to manipulate stored data such as read, write, increment and decrement. Authentication and encryption services are implemented using a proprietary cipher, Crypto1, with a 48-bit key. A number of publications have recently described security vulnerabilities with regards to this product, e.g. [29, 30], mainly focusing on weaknesses identified after the the Crypto1 cipher was reverse engineered [31]. However, since integrity checking is provided using a CRC, parity bits and bit counting the attacker could alter transactions in real time without any knowledge about the encryption algorithm or the secret key material [11]. In theory this attack is very powerful but we were also interested to know whether it is practical and affordable for attackers. To investigate this we implemented this active attack in almost the identical way to the passive attack hardware described in Section 3.1, although the relay communication channel is replaced with a $200 FPGA development board. In our test system, a Mifare Classic 1K token and an off-the-shelf reader with a simple user interface for accessing the token, we can successfully change chosen plaintext bits without being detected. For now we choose to withhold further details about our implementation of this attack as this vulnerability potentially affects a widely deployed product and even though the token appears to exhibit a vulnerability it does not neces-

sarily mean that every system using these tokens is vulnerable. The decision to draw attention to this attack now is to ensure that it is considered by designers looking to upgrade Mifare Classic card systems. Systems could be designed to provide additional measures to prevent bit modification. Possible solutions would be to calculate a Messagce Authentication Code (MAC) and store it on the token along with the data, i.e. data$||\text{MAC}_K(\text{data})$, or to encrypt the data with a block-cipher and to store resultant ciphertext on the token.

5. Relay Resistant Mechanisms

Protecting a system against a passive relay attack is difficult because of the fact that it largely negates application layer cryptography. Additional security measures are therefore needed to supplement existing authentication or encryption mechanisms. In this section we discuss the merits of solutions that have been proposed to detect and prevent relay attacks. We focus on three types of countermeasure:

- Timing Constraints: The attacker's hardware needs time to relay data between the reader and token and the attacker's response is therefore delayed when compared to an authentic response. Implementing time-outs would therefore appear to be a feasible solution to prevent an attacker's 'late' response from being accepted. Timing constraints are already defined for communication in the ISO 14443 standard and readers often have the capability to also implement a time-out on the token's response. The timing constraints in the standards, however, are rarely enforced in readers we observed. Setting a time-out on response data is also not an effective countermeasure as the delay introduced by the relay hardware is much less than the typical time-out values, e.g. the attack hardware in Section 3.1 only caused a 20–35 µsdelay. Setting time-outs that would detect such a small delay is not practical either because the variation in the time taken by the token to generate a response is likely to be larger than the time-out and legitimate responses would be at a risk of being rejected. Further details concerning this countermeasure are given in Section 5.1.

- Distance Bounding: Distance-bounding protocols determine an upper bound for the physical distance between two communicating par-

ties based on the Round-Trip-Time (RTT) of cryptographic challenge-response pairs. The format of the challenge-response pairs are specifically designed to allow for an accurate time measurement, e.g. choosing a response that takes a predictable or constant time to calculate. To achieve an accurate and trusted distance-bound the protocol needs to be run over a special communication channel since it has been shown that conventional channels introduce timing uncertainty that can possibly obscure the delay introduced by a relay attack. Distance-bounding would therefore require modified tokens and/or readers that would increase the total system cost. Distance-bounding has been practically implemented in a contact system but suitable contactless channels are still a work in progress with current proposals raising security or practical concerns. Further details concerning this countermeasure are given in Section 5.2.

- Additional Verification: Relay attacks could be detected or discouraged if additional checking procedures were performed. The token could store a photo of the real token holder that is verified by a human operator, or the token's holder could be requested to enter an additional password or PIN when conducting a transaction. A simple check by a service provider to make sure the the customer is not using a proxy device could also discourage relay attacks. All of these countermeasures, however, complicate the transaction process for the user and/or the service provider. Extra checks also increases the transaction time, which in certain applications is not feasible. When interacting with a reader the token holder could use an additional trusted device to monitor the transaction being conducted with his token. As a result, the holder can ensure that his token is only used in the way intended. This would, however, mean that each holder is issued with an extra hardware devices and would require that the holder is knowledgeable enough to spot any suspicious transactions. Further details concerning this countermeasure are given in Section 5.3.

5.1. Timing Constraints

Practically relaying communication takes additional time and as a result an attacker's response is delayed compared to an authentic response. Strict timing constraints, such as response time-outs, would therefore appear to be a feasible solution for detecting attacks. Unfortunately, timing constraints

are not enforced, with systems accepting responses not adhering to requirements, and if timeouts are strictly enforced these can possibly be bypassed by an attacker. For example, let us consider the timing requirements that the ISO 14443A standard specifies for communication [13].

Timing constraints during the selection and configuration of the token is described in ISO 14443 – Part 3. The reader periodically polls for new tokens using the Request Type A (*REQA*) command. The minimum time between the start bits of two consecutive *REQA* commands is specified as $7000/f_{\text{carrier}} \approx 500$ µs. The token must therefore be able to respond to the *REQA* command with an Answer to Request Type A (*ATQA*) within 5 ms after first receiving an unmodulated carrier. This requirement, however, does not impose an upper bound on the attack delay, since there is nothing linking a specific *REQA* to an *ATQA*. The attacker can simply wait until he has determined the token's response and then answer any of the subsequent *REQA* commands. The standard also specifies a Frame Delay Time (FDT) used to ensure bit synchronization. FDT is specified as $(n \cdot 128 + 84)/f_{\text{carrier}}$ if the last data bit sent by the reader was '1' and $(n \cdot 128 + 20)/f_{\text{carrier}}$ if the last data bit sent was '0'. FDT is calculated using $n = 9$ for *REQA* and *SELECT* commands, and $n \geq 9$ for all other commands. The proxy-token must therefore ensure that the start bit of the response is aligned to a valid FDT value. For $n = 9$ the reader will expect the token's response to start after 91 µs, or 86 µs, depending on the last data bit sent by the reader. This potentially complicates the attack as the real token will only respond at those times, since it thinks that it is speaking to a real reader, which means that the attacker has minimal time to relay the response. However, in our test system the value of n did not really matter as the reader accepted the response as long as the data was aligned to the required bit grid. For the attack to succeed the total response time of the proxy-token, which is the relay delay plus the time taken by the token to respond, therefore has to be set to a multiple of 9.44 µs using the adjustable delay. More information about this experiment and possible reasons for this behaviour is given in [11]. The same conclusion was also drawn by Kasper [12], who determined experimentally that the reader he used accepted responses starting during a time slice of 2.5 µs every 9.44 µs. Even if the n condition was enforced, the attacker could gather the token's information, such as the values of the token's *ATQA* and UID, in advance and respond at the expected time.

After the token has been selected and the communication parameters configured all data exchanges should comply to the timing constraints defined in ISO 14443 – Part 4. The Frame Waiting Time (FWT) specifies the time within which a token shall start its response after the end of the reader's data. FWT is defined as $(256 \cdot 16/f_{\text{carrier}}) \times 2^{FWI}$, where FWI is a value from 0 (FWT = 300 µs) to 14 (FWT = 5 s) with a default of 4 (FWT = 4.8 ms). The value of the Frame Waiting Integer FWI is defined by the token in the *ATS* (Answer To Select) response. If implemented, the Frame Waiting Time defines an upper bound on the relay delay, so in the default case an attacker would need to relay the required data in 4.8 ms. Although implemented timeouts place constraints on the attacker's hardware it does not prevent the attack, and the timeout values are quite long when considering the capabilities of current communication systems. Our attack hardware and the setups discussed in [10, 12] only introduced round-trip delays in the region of 20–35 µs. We therefore believe that it is feasible for an attacker with the necessary resources to implement a relay attack that operates within the 4.8 ms limit. As with the other timing constraints the attacker can also circumvent the shorter timeouts for the *REQA* and *SELECT* commands by getting the token's responses earlier. These values can then be stored in the proxy-token and sent to the reader when required without any delay. Finally, we must also considered the possibility that an attacker could alter data before relaying it back to the reader. For example, an attacker could modify the FWI value transmitted by the token to the maximum value, forcing the reader to implement a 5 second timeout and allowing himself adequate time to relay subsequent data.

Apart from timing constraints specified in the relevant standards the contactless system itself might implement response timeouts. In our test case, the reader's configuration software did allow for a timeout condition to be set for communication between the reader and the token. For the *REQA* and *SELECT* commands the timeout could be set from 300 µs to 76.2 ms, with a default value of 4.8 ms. For any further communication the timeout could be set from 300 µs to 19.7 s, with a default value of 230 ms. These timing measures are vulnerable in the same ways mentioned in the previous paragraph. In contact systems the timing constraint are just as tolerant to high latency. In [9] the authors state that they could add an additional 3 s delay to the base time their hardware took to relay communication and still succeed with their attack. Even short, high-resolution, timeouts do not

provide an acceptable solution as variability in the processing time taken to generate a response could lead to inaccuracies, e.g. if the token takes 100 ms $\pm$ 1% the 1 ms uncertainty possible allows enough time for relaying, while it is possible that an attacker can gain a timing advantage by overclocking a token and receiving a response early, as demonstrated in [32]. As a result we must conclude that the specified timeouts and timing constraints currently defined in current standards or hardware cannot provide adequate protection against relay attacks.

5.2. Distance Bounding

Distance-bounding protocols determine an upper bound for the physical distance between two communicating parties based on the Round-Trip-Time (RTT) of cryptographic challenge-response pairs. This distance can then be used as a cryptographic proof of proximity. Secure distance-bounding protocols are meant to detect any extra delay in the prover's expected response. These protocols, if implemented correctly, can therefore be an effective way to prevent relay attacks. Brands and Chaum [33] first described distance-bounding protocols in 1993 and several new protocols have been proposed since, e.g. [34, 35, 36]. The design of distance-bounding protocols is beyond the scope of this paper but to give the reader an indication of how these protocols function we briefly discuss the protocol by Brands and Chaum, as shown in Figure 6. For a more detailed overview of different distance-bounding protocols please see [11, Chpt 5]

In the setup stage the verifier and the prover both generate a random bit string of length l, which serves as the response string M and the challenge C, respectively. Making the prover calculate the responses R based on a response string M and the challenges C, forces the prover to wait until he received the challenge before transmitting his response. The protocol further prevents distance fraud by specifying that the prover commits to the string M before the exchange phase starts. This prevents the prover from sending a random bit R_i before receiving C_i, and then retrospectively claiming during the verification stage that he used $M_i = C_i \oplus R_i$. The verifier then transmits one challenge bit C_i at a time (for all$i = 1, \ldots, n$), to which the prover responds immediately with $R_i = C_i \oplus M_i$. The verifier times the round-trip delay Δt_i between sending each bit C_i and receiving the corresponding response bit R_i. During verification the prover reveals M and transmits a digital signature, or message authentication code, of the two bit strings C and R. This allows the verifier to check whether the prover received the challenge

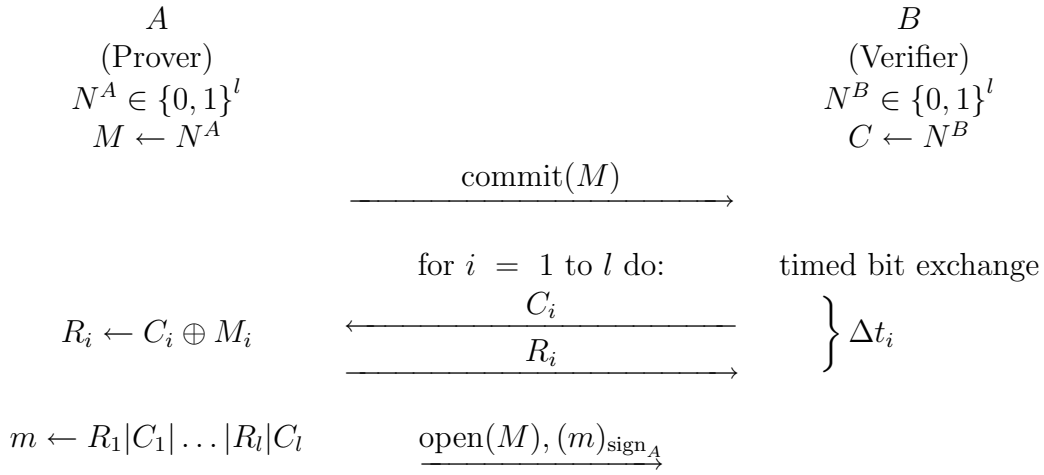


Figure 6: Brands-Chaum distance-bounding protocol

bits it sent. This prevents an attacker from sending guessed challenges C_i' to the prover and recovering M, from the received responses by calculating $M = C' \oplus R$, and subsequently using the recovered response string once the verifier starts the exchange stage. For this attack to succeed the attacker would therefore need to guess all of C correctly. The verifier also checks that $M_i = C_i \oplus R_i$ for $i = 1$ to l, thus confirming that the prover sent the right response and used the string M he committed to.

Even if the cryptographic part of a distance-bounding protocol is secure the design should also take into consideration the practical aspects of the prover calculating the response and the communication channel. In general, distance-bounding protocols will use short data formats, e.g. single bits, and simple response calculations, e.g. XOR and 1-bit lookup, to achieve more secure and accurate timing information. Proposals requiring the prover to perform excessive computation upon receipt of the challenge to generate the response could yield unreliable timing information due to variability in the processing time. Such proposals are basically an authentication protocol with a tight timeout constraint, and therefore susceptible to similar weaknesses as normal timeouts. .

Time-of-flight distance-bounding protocols are dependent on time measurements made at the physical layer of the communication channel to accurately calculate the distance between the prover and verifier. This means that the security of the distance bound depends not only on the cryptographic

protocol itself but also on the practical implementation and the physical attributes of the communication channel. The communication channel used for the exchange must, therefore, not introduce any latency that the attacker can exploit to circumvent the physical distance bound. Clulow *et al.* [37] show how an attacker can gain a timing advantage by exploiting the time allowed by the verifier for the transmission of redundant data, such as framing and error correction, at the packet level of the communication layer. For example, if a challenge is followed by a CRC the attacker does not wait until he receives the entire data packet before forwarding it to the proxy-reader. Instead the attacker only forwards the challenge the moment is has been received, and the proxy-reader calculates and appends the CRC before sending it to the token. Assuming the proxy-reader can calculate the CRC in time t_x and the reader takes time t_y to transmit the CRC the distance bound will not detect the attack if the communication is relayed to the proxy-reader and back in less than $t_y - t_x$. Hancke *et al.* [32] also demonstrated how the attacker can achieve similar timing benefits at the physical level, i.e. by exploiting time delays in the coding and modulation stages of RF transceivers. Both these papers illustrate that systems planning to use distance-bounding protocols must implement special low-latency channels as conventional communication channels are designed for reliable data transfer and therefore feature redundancy and timing tolerances, which introduces timing uncertainty for an attacker to exploit.

Considering these weaknesses, the implementation of a suitable distance-bounding channel for contactless tokens is a technical challenge. Currently, there are two basic ideas proposed in literature. The first approach is to work with the current communication channel principles and try to add distance-bounding functionality. There are two proposals tailored to the HF contactless environment where the verifier directly samples the modulated carrier, which results in the prover's response being recovered without performing traditional demodulation and decoding, thus reducing communication channel latency. In the proposal by Munilla, et al. [35], the reader transmits a periodic sequence of pulses that are 100% amplitude-shift key (ASK) modulated onto the carrier. The pulses act as synchronization bits with the periods in between, when the carrier is off, referred to as slots. In some slots, the reader will switch on the carrier for a short period of time to indicate that it wants a response. The token knows when to expect these requests and preemptively switches its impedance to indicate the answer. When the

reader then switches on the carrier, the envelope of the signal rises immediately to a level that indicates the token's answer state. The reader times from the point when it switches on the carrier until the token's response can be determined. To determine the token's response, the reader continiously samples the envelope of the carrier until it finishes rising and becomes stable. The time it takes until the two levels can be reliably distinguished, and the difference between the envelope amplitude for the two states, depends on the distance between the token and the reader. The authors state that the timing resolution of the channel is less than 1 µs. For this proposal to be secure the token would need to be protected against a proxy-reader transmitting a weak carrier, which appears to the token to be 'off', to probe the state of the load early. Another practical drawback is that the carrier is switched off regularly, which means that the token has no source of power for long periods of time. A proposal by Reid, et al. assumes that the token will reply after a fixed time t_{wait} [36]. In practice the token waits for a pre-determined number of cycles of the 13.56 MHz carrier, which would synchronise its response to an accuracy of 1/13.56 MHz = 75 ns. The reader then make a distance-bounding estimate based on the difference between the time it expected the response, t_{wait}, and when the actual time that the response was detected. The system tries to determine the exact moment that the amplitude of the carrier is first modulated by sampling the peaks of the HF carrier and comparing the latest sample to a threshold calculated from the eight previous samples. The resolution of the system is once again dependent on the distance between the token and the reader, with the authors stating that a 300 ns resolution was obtained when the token and the reader were 4–5 cm apart. A possible vulnerability is the authors' stated assumption that the token should be protected against overclocking, which is not always feasible. Both contact and contactless tokens receive their clock signals from the reader so an attacker could use a proxy-reader to provide a token with higher frequency clock. As demonstrated in [32], this could cause the token to process data at a faster rate, which shortens the token's wait time t_{wait} and therefore causes the token to respond earlier than expected. As the attacker learns the response earlier he has time to relay this response to the proxy-token, which can now reply the reader at the expected time.

The second approach advocates that a new channel is implemented using a crude Ultra-Wideband (UWB) pulse channel [34]. Making the bit period as short as possible would limit the attacks, although this requirement might

compromise the reliability of the channel. The channel is therefore only used for timed challenge-response exchanges and the distance-bounding protocol is designed in such a way that bit errors can be tolerated. Due to the fact that tokens are generally susceptible to overclocking [32], since they receive a clock from an external source, it is also proposed that the response should be calculated using asynchronous logic. Using the 13.56 MHz carrier for loose synchronization, the reader starts timing on the zero-crossing of the carrier, waits for t_{t}, and then transmits the challenge bit C_i. The token also waits for the zero-crossing of the carrier before it starts the sampling process. The sampling time t_{s} is fixed and dependent on the token's hardware implementation. The reader tries to ensure that the token samples C'_i correctly by adjusting delay $t_{\mathrm{t}} \approx t_{\mathrm{s}}$, essentially aligning the challenge bit period with the time the token samples. By varying the delay t_{t} during the first few values of i until a delay has been found that results in the correct response bits, the reader can adjust itself automatically to any component tolerances and instabilities that may affect the exact sampling time in the token. After a brief processing delay t_{d} the prover transmits a response bit R_i. After time t_m the reader samples the channel to determine R'_i. The propagation time t_p can then be calculated by the reader as follows: $t_{\mathrm{p}} = (t_{\mathrm{m}} - t_{\mathrm{t}} - t_{\mathrm{d}})/2$. The accuracy of the timing measurement is dependent on the width of the pulses and the hardware capabilities of the verifier. Although this approach appears to be theoretically more secure than the other two proposals it is yet to be practically demonstrated in a contactless environment.

Drimer *et al.* [9] practically implemented distance bounding for contact systems using a wideband pulse approach similar to that mentioned in the previous paragraph. Their distance bounding scheme was successfully implemented and tested on an FPGA development system to detect 2.0, 1.0, and 0.3 meter transmission lengths, although it can be modified to work for any distance. Although the hardware requirements are not excessive, and existing clock and data lines are utilized, implementing this scheme would nevertheless require modifications to be made to both the terminal and the token, which would result in increased system cost. As the authors note, additional 'costs', such as added time taken per transaction, might also be incurred when integrating distance bounding into current payment systems. The main advantage of this countermeasure is that it is transparent to the service providers and the token's holder. In other words, the service provider and the token holder are not inconvenienced by having to perform extra

duties related to security as the distance-bounding procedure is handled automatically by the reader and token.

5.3. Additional Verification Procedures

In certain cases, relay attacks could be detected if additional procedures were incorporated into token transactions. Systems that require additional verification of the holder, either through human validation or by implementing 'two factor' authentication (2FA), could therefore limit the success of this attack. An attacker, for example, would struggle to execute the attack against RFID enabled e-passports since the photo read from his 'passport' does not resemble him. Simple physical checks on the 'token' presented would also force the attacker to construct a proxy-token that not only functions like a real token but also looks like one. Unfortunately these measures are not always practical. A benefit often attributed to contactless systems is speed. Asking a queue of travelers to enter their PINs, or checking a photo stored on their token, adds unwanted delay, so additional verification is not a reasonable mechanism in such systems. 2FA is a more favourable solution for access control systems, and should already be implemented to prevent unauthorised access with stolen or lost tokens. It must, however, be said that a second authentication factor does not necessarily prevent the attack, as illustrated by the attack implementation described in Section 3.2, but does complicate the attack procedure. Another of the perceived benefits of contactless systems is convenience. Contactless payment systems expect the customer to quickly hold a wallet, or purse, to a reader and operators asking to inspect the token could potentially irritate a customer and again cause delays for other token holders. Even in contact payment systems the attacker tends to keep control over his token at all times, and the merchant might even look away to allow the customer to enter his PIN [9].

Another method that has been proposed is that the legitimate holder should verify that his token is used in an approved way, e.g. checking whether the amount awaiting authorisation is as expected, with the aid of an additional device. One example of this approach is the use of an electronic attorney as described in [38]. The electronic attorney is an additional hardware device owned by the token holder that would act as an intermediary between the token and reader. The device would provide a trusted interface that displays details of the transactions between the legitimate token and reader to the token holder. In this case the token holder does not have to solely rely

on information displayed by the reader so he can verify that the merchant's reader and his token are acting in an honest way. An extra trusted device offers several advantages, especially in payment systems, but could in turn lead to increased costs and operating complexity and possible inconvenience to the customer.

6. Conclusion

Smart tokens are often used in proximity identification systems. These systems assume that the token is in close proximity to the reader if it is authenticated because of the perceived physical limitations of the communication channel. A relay attack exploits this assumption and allows an attacker to temporarily possess a 'virtual clone' of a legitimate token, which enables him to gain access to associated benefits. Even thought the basic attack theory was first proposed several decades ago and 'worm holes' are a recognised security threat in wireless networks, few threat models, or operational guidelines, for contact and contactless smart tokens take it into account. This paper provides a detailed overview of relay attacks with regards to smart tokens during which we highlight their practical feasibility and provide examples of how current systems are potentially vulnerable. We hope that this paper will help the reader to recognise the threat posed by relay attacks and understand the challenges related to preventing or detecting these attacks.

The theory of passive and active relay attacks are explained and some examples of practical relay attacks that have been demonstrated are examined in further detail. A description is given of an attack that we successfully implemented and demonstrated against an ISO 14443A contactless system using guidelines in public literature and easily obtainable hardware. A similar demonstration concerning contact tokens, as presented in [9], is also briefly discussed. We also present some exploitation scenarios for relay attacks against current systems, including a novel attack against NXP's Mifare Classic tokens. This active relay attack, which we successfully implemented against a test system, allows an attacker to potentially circumvent system security without any key recovery or knowledge of the Mifare Classic's Crypto1 algorithm. The motivation for highlighting the attack at this time, is to ensure that it is considered during evaluation and migration planning of current

smart card systems.

A relay attack is difficult to defend against as it successfully circumvents application layer security mechanisms. We examine a number of countermeasures that have been proposed. Timing constraints are shown to be ineffectual and although two factor authentication can be used in certain systems it nullifies some key advantages of smart token systems, such as speed and convenience. Distance bounding and allowing the token's holder to monitor transaction details using a trusted interface may provide better protection but both solutions could add significant cost and/or complexity to existing systems. As a result of these current disadvantages there are still future scope for research on appropriate security mechanisms.

References

[1] J.H. Conway. *On Numbers and Games*. Academic Press, 1976.

[2] Y.C. Hu, A. Perrig and D.B. Johnson. *Wormhole attacks in wireless networks.* IEEE Journal on Selected Areas in Communications (JSAC), pp 370–380, 2006.

[3] A. Juels. *RFID Security and Privacy: A Research Survey*. IEEE Journal on Selected Areas in Communications, Vol. 24, Issue 2, pp 381–394, February 2006.

[4] P. Rotter. *A Framework for Assessing RFID System Security and Privacy Risks*. IEEE Pervasive Computing Magazine, Vol. 7, Issue 2, pp 70–77, April 2008.

[5] C.M. Roberts. *Radio Frequency Identification (RFID)*. Elsevier Journal Computers & Security, Vol. 25, Issue 1, pp 18–26, February 2006.

[6] Bundesamt für Sicherheit in der Informationstechnik. *Security Aspects and Prospective Applications of RFID Systems*. October 2004. `http://www.bsi.bund.de`

[7] NIST: Special Publication 800-98. *Guidance for Securing Radio Frequency Identification (RFID) Systems*. April 2007. `http://csrc.nist.gov/`

[8] Y. Desmedt, C. Goutier and S. Bengio. *Special Uses and Abuses of the Fiat-Shamir Passport Protocol*. Advances in Cryptology (CRYPTO), Springer-Verlag LNCS 293, pp 21, 1987.

[9] S. Drimer and S.J. Murdoch. *Keep Your Enemies Close: Distance Bounding Against Smartcard Relay Attacks.* Proceeding USENIX Security Symposium, pp 87–102, August 2007.

[10] G.P. Hancke. *A practical relay attack on ISO 14443 proximity cards.* `http://www.rfidblog.org.uk/hancke-rfidrelay.pdf`

[11] G.P. Hancke. *Security of Proximity Identification Systems*. PhD Dissertation, University of Cambridge, February 2008.

[12] T. Kasper. *Embedded Security Analysis of RFID Devices*. Diploma Thesis, Ruhr-University Bochum,
July 2006.

[13] ISO/IEC 14443. *Identification cards – Contactless integrated circuit cards – Proximity cards.*

[14] *OpenPCD* Project. `http://www.openpcd.org`

[15] ISO/IEC 18092. *Information technology – Telecommunications and information exchange between systems – Near Field Communication – Interface and Protocol (NFCIP-1).*

[16] Z. Kfir and A. Wool. *Picking virtual pockets using relay attacks on contactless smartcard systems.*
Proceedings of IEEE/CreateNet SecureComm, pp 47–58, 2005.

[17] ISO/IEC 15693. *Identification cards – Contactless integrated circuit cards – Vicinity cards.*

[18] ISO/IEC 18000. *ISO/IEC 18000 Information Technology AIDC Techniques-RFID for Item Management – Air Interface.*

[19] EPC Class-1 Generation-2 UHF RFID Conformance Requirements Specification v. 1.0.2

[20] Texas Instruments. *HF Antenna Design Notes, Technical Application Report.*
http://www.ti.com/rfid/docs/manuals/appNotes/HFAntennaDesignNotes.pdf

[21] I. Kirschenbaum and A. Wool. *How to Build a Low-Cost, Extended-Range RFID Skimmer.* Proceedings of 15th USENIX Security Symposium, pp 43–57, August 2006.

[22] K. Finkenzeller. *RFID Handbook: Radio-frequency identification fundamentals and applications.*
2nd Edition, Wiley, 1999.

[23] EMV Integrated Circuit Card Specifications for Payment Systems, v4.1. June 2007.
http://www.emvco.com/

[24] VISA Internation Service Association. Approved PIN entry devices, June 2008. http://partnernetwork.visa.com/dv/pin/pedapprovallist.jsp

[25] N. Borisov, I. Goldberg and D. Wagner. *Intercepting Mobile Communications: The Insecurity of 802.11.* Proceedings of Seventh Annual International Conference on Mobile Computing and Networking,
pp 180–189, July 2001.

[26] NXP Semiconductor. *Mifare Standard Card IC MF1 IC S50 Functional Specification.*
http://www.nxp.com/acrobat_download/other/identification/m001051.pdf

[27] NXP Semiconductor. *Mifare Standard Card IC MF1 IC S70 Functional Specification.*
http://www.nxp.com/acrobat_download/other/identification/m043531.pdf

[28] Mifare.net. http://mifare.net/

[29] N.T. Courtois, K. Nohl and S. O'Neil.*Algebraic Attacks on the Crypto-1 Stream Cipher in MiFare Classic and Oyster Cards.* Cryptology

ePrint Archive: Report 2008/166, April 2008.http://eprint.iacr.org/2008/166

[30] Security Flaw in Mifare Classic. Radboud University Nijmegen, March 2008.
http://www.ru.nl/english/general/radboud_university/vm/security_flaw_in/

[31] K. Nohl, D. Evans, Starbug and H. Plötz. *Reverse-Engineering a Cryptographic RFID Tag*. Proceedings of the 17th Usenix Security Symposium, August 2008.

[32] G.P Hancke and M.G. Kuhn. *Attacks on Time-of-Flight Distance Bounding Channels*. Proceedings of the First ACM Conference on Wireless Network Security (WISEC'08), pp194–202, March 2008.

[33] S. Brands and D. Chaum. *Distance Bounding Protocols*. Advances in Cryptology, EUROCYPT '93, Springer-Verlag LNCS 765, pp 344–359, May 1993.

[34] G.P Hancke and M.G. Kuhn. *An RFID distance bounding protocol*. Proceedings of IEEE/CreateNet
SecureComm, pp 67–73, September 2005.

[35] J. Munilla, A. Ortiz and A. Peinado. *Distance Bounding Protocols with Void Challenges for RFID*.
Proceedings of Workshop on RFID Security (RFIDSec), pp 15–26, July 2006.

[36] J. Reid, J.M.G Nieto, T. Tang and B. Senadji. *Detecting Relay Attacks with Timing-Based Protocols*. Proceeding 2nd ACM Symposium on Information, Computer and Communications Security, pp 204–213, March 2007.

[37] J. Clulow, G.P. Hancke, M.G. Kuhn, T. Moore. *So Near and Yet So Far: Distance-Bounding Attacks in Wireless Networks.* European Workshop on Security and Privacy in Ad-Hoc and Sensor Networks (ESAS), Springer-Verlag LNCS 4357, pp 83–97, September 2006.

[38] R.J. Anderson, and M. Bond. *The Man-in-the-Middle Defense*. Presented at Security Protocols Workshop, March 2006. http://www.cl.cam.ac.uk/~rja14/Papers/Man-in-the-Middle-Defence.pdf