Technical Report

UCAM-CL-TR-752
ISSN 1476-2986

Number 752

Security of proximity identification systems

Gerhard P. Hancke

July 2009

This technical report is based on a dissertation submitted February 2008 by the author for the degree of Doctor of Philosophy to the University of Cambridge, Wolfson College.

ISSN 1476-2986

Security of Proximity Identification Systems

Gerhard P. Hancke

Summary

RFID technology is the prevalent method for implementing proximity identification in a number of security sensitive applications. The perceived proximity of a token serves as a measure of trust and is often used as a basis for granting certain privileges or services. Ensuring that a token is located within a specified distance of the reader is therefore an important security requirement. In the case of high-frequency RFID systems the limited operational range of the near-field communication channel is accepted as implicit proof that a token is in close proximity to a reader. In some instances, it is also presumed that this limitation can provide further security services.

The first part of this dissertation presents attacks against current proximity identification systems. It documents how eavesdropping, skimming and relay attacks can be implemented against HF RFID systems. Experimental setups and practical results are provided for eavesdropping and skimming attacks performed against RFID systems adhering to the ISO 14443 and ISO 15693 standards. These attacks illustrate that the limited operational range cannot prevent unauthorised access to stored information on the token, or ensure that transmitted data remains confidential. The practical implementation of passive and active relay attacks against an ISO 14443 RFID system is also described. The relay attack illustrates that proximity identification should not rely solely on the physical characteristics of the communication channel, even if it could be shown to be location-limited. As a result, it is proposed that additional security measures, such as distance-bounding protocols, should be incorporated to verify proximity claims. A new method, using cover noise, is also proposed to make the backward communication channel more resistant to eavesdropping attacks.

The second part of this dissertation discusses distance-bounding protocols. These protocols determine an upper bound for the physical distance between two parties. A detailed survey of current proposals, investigating their respective merits and weaknesses, identifies general principles governing secure distance-bounding implementations. It is practically shown that an attacker can circumvent the distance bound by implementing attacks at the packet and physical layer of conventional communication channels. For this reason the security of a distance bound depends not only on the cryptographic protocol, but also on the time measurement provided by the underlying communication. Distance-bounding protocols therefore require special channels. Finally, a new distance-bounding protocol and a practical implementation of a suitable distance-bounding channel for HF RFID systems are proposed.

Acknowledgments

I would like to thank my supervisor Markus Kuhn for his assistance and guidance on a variety of research and non-research issues during my time at the Computer Laboratory. In addition, his help with publications and comments on this dissertation was invaluable. I am also grateful to Saar Drimer for reading much of my earlier drafts and being available for meaningful discussion. Sergei Skorobogatov's assistance with the decapsulation of some RFID tokens and Nicholas Cutler's effort in tracking down a copy of the NIST report on HF RFID eavesdropping are much appreciated. I also wish to thank everyone who contributed to my Cambridge experience over the last four years, especially Lisa, Mat, John, Ulrich and the Wolfson College community in general. For the entire duration of my study I was generously supported by the Skye Foundation, without whom all of this would not have been possible.

Contents

Chapter 1

Introduction

RFID technology is the prevalent method for implementing proximity identification in a number of security sensitive applications. Ensuring that a token is located within a specified distance of the reader is therefore an important security requirement. In the case of HF RFID tokens, also known as proximity or contactless cards, it is sometimes assumed that the limited operational range of the near-field communication channels provides security services such as data confidentiality, authentication and proof of proximity. In recent years, however, there have been questions raised about the belief that near-field channels are strictly location-limited. This dissertation examines the security of current proximity identification systems using RFID devices and investigates the use of distance-bounding protocols to make these systems more secure.

One of the significant security concerns with RFID devices is the possible leakage of personal information, or location, to unauthorized parties. Two obvious ways in which an unauthorized party can get information are skimming and eavesdropping. Skimming occurs when the attacker uses his reader to access information on the victim's RFID token without consent. Eavesdropping occurs when the attacker intercepts communication between an RFID token and an authorized reader. To avoid detection the attacker would need to execute these attacks at a distance greater than the expected operational range, in effect circumventing the location-limited nature of the near-field channel. The distances at which these attacks are possible are often debated and used as an indication of RFID security. Despite this, there are few published results describing practical attack experiments and results. Determining how these attacks affect current systems makes an important contribution toward formulating a more accurate threat model for RFID systems.

RFID systems operating on the assumption that the token is in close proximity to the reader, because of the physical limitations of the communication channel, are potentially also vulnerable to an attack where the attacker relays communication between the reader and a token over a greater distance than intended. A successful relay attack allows an attacker to temporarily possess a 'clone' of a token, thereby allowing him to gain the associated benefits. Whereas skimming and eavesdropping attacks can be limited by appropriate authentication and encryption mechanisms, relay attacks are not really affected by application layer security since the attacker never needs to know the plain-text data, or any secret key material, as long as he can continue relaying the respective messages.

Distance-bounding protocols provide a cryptographically verified upper bound on the distance between two devices, the verifier and the prover. These protocols are integrated into the underlying communication channel and are meant to detect any extra delay in the prover's expected response. To my knowledge this is the only cryptographic way of preventing relay attacks. These protocols are also useful as building blocks in secure location systems.

Distance-bounding protocols execute a timed cryptographic exchange. The round-trip-time measurement is then used to make a distance estimate based on the propagation time of the underlying communication medium. In the last few years there have been a number of proposals for new distance-bounding protocols, each with its own merits and weaknesses. Currently, most of these protocols are not suited to the RFID environment. Low-resource tokens might be unable to implement the required cryptographic primitives and transferring long data sequences over a channel with relatively low data rate is also not ideal for systems where the transaction time is limited.

Since a distance-bounding protocol bases its distance estimate on round-trip-time, it must be implemented using a communication channel that will provide an accurate and secure time measurement. This suggests that the security of the distance-bounding protocol depends not only on the cryptographic protocol itself but also on the practical implementation and the physical attributes of the communication channel. For this reason the communication channel used for the exchange must not introduce any latency that the attacker can exploit to circumvent the physical distance bound. Unfortunately the implications of the communication channel's implementation on the security of the distance bound have not been discussed yet in the literature, despite its importance. It would, therefore, be useful to investigate whether an attacker could circumvent a distance bound by exploiting weaknesses at the packet level, i.e. cryptographic exchange and frame formatting, or physical layer, i.e. coding and modulation, of the underlying communication channel.

1.1 RFID security and privacy

RFID tokens do not need physical contact with a reader, which means that a transaction can be conducted while the token is still in the user's pocket or purse. This feature simplifies operation and increases the overall transaction speed, i.e. the user does not need time to take the token out or align it in a specific way. This lack of human interaction during a transaction has, however, led to fears that the information stored on RFID tokens could be acquired without the user's consent. Each token has a unique identification number that could be used to track its owner. As a result it is a possibility that large RFID deployments could profile and track individual users. As a result, various groups are concerned about the 'big brother' potential of RFID technology [1, 28, 142]. As RFID tokens are used for transactions of increasing value, they could also become the target of individual attackers, who, if able to gain access to the information on the RFID token, might be able to engage in an act of 'digital pick-pocketing' while just standing next to the victim. These scenarios have generated significant interest in RFID security and there are numerous publications dealing with this theme.

Any related work pertinent to this dissertation is provided in the chapter to which it is applicable. This section very briefly discusses some RFID security topics that are not directly addressed in detail within this dissertation.

Due to the wide scope of the RFID security research field I only provide a limited number of examples:

- Physical Security: The simplest approach for providing privacy is to disable the token once the associated product is transferred to the user. This can be done by removing or destroying the token. A number of tokens also implement a 'kill' command [50]. Upon receiving this command from the reader the token renders itself inoperable. Alternatively a token could be shielded from radio signals by enclosing it in a Faraday cage consisting of a metal mesh or foil.
- Lightweight Cryptography: Most publications describe security mechanisms for low-resource tokens that protect the user's privacy and prevent unauthorised access to the token. It is assumed that *basic* tokens, used mostly for item management, contain very limited cryptographic primitives. Simple privacy suggestions for these RFID tokens include the use of pseudonyms [82], intelligent relabeling [144] or modified anti-collisions protocols [168] to disguise the unique identifier of the tokens. *Symmetric tokens* have the ability to compute a pseudo-random or cryptographic one-way function so they can implement additional mechanisms, e.g. hash-locking.
- Proxies and Blocking: The user may choose to carry an additional device that ensures privacy. Such a device can enforce a comprehensive privacy policy by acting as the intermediary between the reader and token [141] or simply choose to block certain bits during anti-collision, which will prevent a reader from obtaining the entire unique identifier [85].
- Hardware Security: There are various papers that describe optimised implementations of existing cryptographic algorithms that are claimed to be suitable for low-resource devices, e.g. [52]. There are a very limited number of practical attack papers that describe topics such as reverse engineering [11] and side-channel analysis [26, 72].

For a detailed overview of research in RFID security please consult the relevant survey by Juels [83] and the comprehensive list of academic papers on the topic, maintained by Avoine [4].

1.2 Motivation and scope

RFID security has become a popular topic in the past few years as the result of the technology's increased deployment. Privacy is one of the main concerns, with consumers worrying that they can be tracked by large companies or that their personal details, if contained on an RFID device, could be leaked to an unauthorised party. A few months before I started my PhD with the Computer Laboratory's Security Group both Walmart and Tesco cancelled RFID trials with Gillette amid customers' protest [36, 163]. At the time HF RFID tokens were also being considered for implementation in security sensitive applications such as machine readable travel documents and open payment systems, raising concerns about unauthorised reading and eavesdropping, e.g. [127]. This resulted in a lively debate about the possibility of an attacker actually implementing these attacks to covertly obtain information stored on a token, especially when taking into account the limited operational range of near-field communication. Despite the public interest not much work had been done on studying the feasibility of these attacks. Implementing these attacks and documenting actual results and experimental procedures were therefore required to increase understanding of the RFID threat model.

I became interested in all aspects of RFID security, as described in Section 1.1, although my research focused on the assumption that the limited operational range of HF RFID systems could be used to provide security services. I therefore investigated to what extent near-field communication could be seen as location-limited, which led to work on how an attacker could extend the communication limits by implementing eavesdropping, skimming and relay attacks. Since I was already interested in practical hardware security I decided to try and implement proof-of-concept attacks exploiting the radio communication interface of HF RFID devices. This work also combined two areas in which our group already had a research history, compromising emanations and smart card security, so I already had some equipment available, e.g. antenna set and RF receiver for measuring electromagnetic signals. I decided to concentrate on proximity tokens since they are often implemented in secure applications, like e-passports and contactless credit cards, which would provide additional incentives for attackers. I believed that a thorough practical threat analysis would contribute toward the general security analysis of contactless cards and aid in future security policy formulation, which would make systems more attractive to industry and consumers. My work on eavesdropping and skimming attacks is presented in Chapter 3. My work on relay attacks is presented in Chapter 4.

Relay attacks are especially interesting since they can be implemented even if the communication was shown to be location-limited. These attacks also circumvent most application layer security mechanisms and as a result additional security mechanisms are needed to detect these attacks. Reading the "Distance-Bounding Protocols" paper by Brands and Chaum [13] prompted my interest in distance bounding and generating cryptographic proximity proofs. Having already demonstrated that RFID systems were vulnerable to relay attacks, I wanted to investigate the possibility of using distance-bounding protocols in the RFID environment to provide secure proximity identification services.

Several distance-bounding protocols have been published in recent years. To implement distance bounding for RFID systems, each of these protocols' merits and weaknesses would need to be analysed in order to decide on a suitable protocol to implement. An overview of distance-bounding protocols and a new proposal tailored to proximity RFID systems are presented in Chapter 5. As work on distance-bounding protocols progressed it became clear that an attacker could exploit latency at the packet and physical layer of the communication channel to circumvent the distance bound. Special care must therefore be taken to use suitable channels that will not compromise the security of the cryptographic protocol. The implementation of distance-bounding channels are discussed in Chapter 6.

In Chapters 3 to 6 I assume that the reader has some basic knowledge of RFID systems. Chapter 2 is intended as a primer for readers unfamiliar with RFID technology, near-field communication principles and the relevant industry standards.

Chapter 2

Radio frequency identification

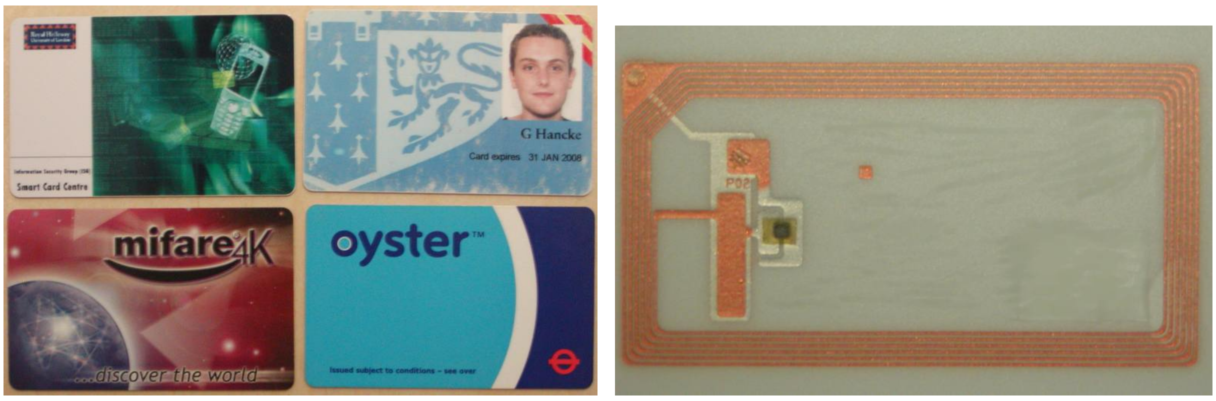
An increasing number of 'contactless' systems are based on passive Radio Frequency Identification (RFID) technology. A passive RFID token is powered by a transmitted RF carrier, which is also used for bi-directional communication. RFID technology comprises of several standards, which are suitable for different applications. Electronic Product Code (EPC) tags, contactless credit cards, e-passports and access control are just a few examples of systems that use a subset of this technology. This chapter contains a brief explanation of RFID operating principles along with an overview of prominent implementations and industry standards.

2.1 Introduction

RFID is a technology that increases productivity and convenience and is currently being integrated into many areas of society. RFID is flexible and the variety of standards and tokens available allow it to be tailored to any application. The technology also offers additional benefits such as reduced maintenance cost and extended product lifetime. The uses of Radio Frequency Identification have therefore grown remarkably and it is believed that 1.3 billion tokens were sold in 2006. The total RFID market, including systems and services, is valued at £1.4 billion and expected to increase to £13 billion by 2016 [40].

Despite its recent popularity, RFID technology has been around for more than half a century. It is commonly believed that the concept of radio identification started in the early 1940s with the advent of radar when IFF (Identification Friend or Foe) transponders actively modulated the radiated ground radar signals to identify airplanes. Despite early work on RFID, such as Stockman's "*Communication by Means of Reflected Power*" in 1948 [157], recognizing the potential of RFID, it took several more years, and additional advances in electronics, before the technology was implemented in further applications. During the 1960s, stores and libraries used electronic article surveillance, an early 1-bit form of RFID, for theft control. Meanwhile private and government research on the subject continued and in 1973 the first patents that resembled modern systems were filed: a token with rewritable memory by M.W. Cardullo and a passive token used to unlock doors by C. Walton. Scientists from the Los Alamos National Laboratory, who were asked by the U.S. Energy Department to develop a tracking system for nuclear materials, also demonstrated the concept of modulated backscatter with 12-bit tokens operating at 915 MHz in the same year. The basic communication principle of this system is still used today by the majority of Ultra-High Frequency (UHF) RFID tags. In the late 1980s, RFID gained widespread acceptance in automated toll collection and access control systems, which was followed by implementation in public transport payment systems and the first serious attempts at standardization in the 1990s. In 1999 it was proposed that low-cost UHF RFID ‘tags’ could be used to track items in supply chains. Currently the use of Electronic Product Code (EPC) tags for tracking at the pallet, case and item level is probably one of the most prominent RFID application, driven by government agencies, such as the US Department of Defense, and various large retailers, such as Tesco and Wal-Mart. RFID technology is also used on a large scale in other applications such as machine readable travel documents (e.g. e-passports), ticketing, access control and payment [100, 135, 140].

This chapter is intended as an introduction to RFID that summarizes the aspects most relevant to proximity identification systems. In Section 2.2 I give a brief overview of existing systems, describing the general operating principles and available technology before discussing a number of high-profile implementations. I also discuss the RF interface and communication theory in Section 2.4 before providing a summary of the current High Frequency (HF) RFID standards in Section 2.5. I recommend that anybody wishing to learn more about the subject of RFID should consult literature by the Smart Card Alliance [150] and the “*RFID Handbook*” by Klaus Finkenzeller [54]. A number of open source RFID projects also provide hardware and software that can facilitate better understanding by means of practical experimentation [126, 132, 134].



(a) Example of tokens

(b) An example of an inlay, showing RFID IC and antenna

Figure 2.1: ‘Contactless’ tokens

2.2 ‘Contactless’ technology

Even though RFID is a collective term for a number of technologies, it is often used primarily to describe applications using low-resource devices, such as EPC tags. Devices used in identity and payment systems, like the cards shown in Figure 2.1(a), are instead commonly referred to as ‘contactless’ or ‘proximity’ tokens. These devices operate in the HF radio band, contain more resources and have a much shorter operating range than their UHF counterparts.

The basic operation of a contactless system is shown in Figure 2.2. Each token, or Proximity Integrated Circuit Card (PICC), contains an antenna and an integrated circuit (IC) as shown in Figure 2.1(b). The IC performs modulation and demodulation of the RF channel and is also responsible for data storage and processing. The passive token derives its power from the RF carrier transmitted by the reader, or 'Proximity Coupling Device' (PCD). The bi-directional communication between the token and reader is also modulated onto this carrier.

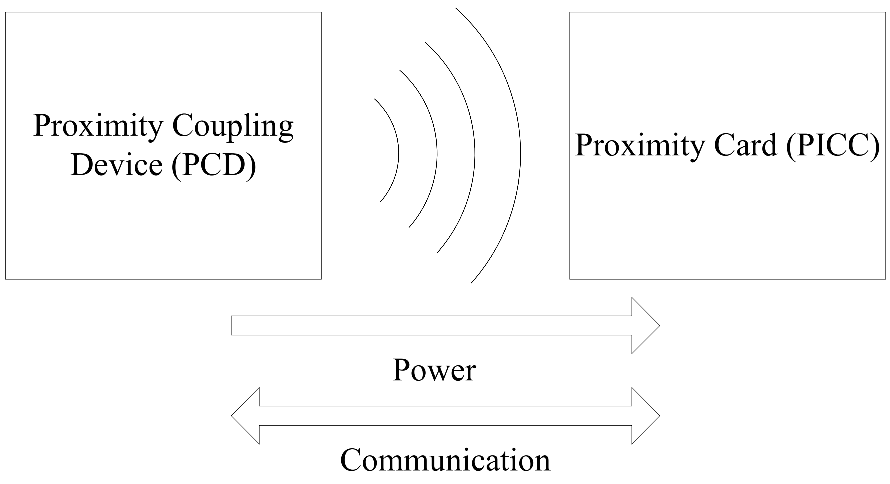


Figure 2.2: Basic operation of an RFID system

The main benefit of contactless technology is its ease of use. The user does not have to physically insert his token into the reader, or orientate the token in a specific way. In most cases the token can be kept in a wallet or a purse, providing some measure of personal security. All these factors combine to provide fast transactions and ensure high throughput. Furthermore, readers and tokens have no external mechanical parts that can wear out. This makes systems more durable and reliable, especially in exposed or dirty environments, and reduces maintenance costs when compared to contact or magnetic stripe systems. The fact that the token does not need to contain a power supply adds to the lifetime of the token as well. Contactless tokens can also provide the same security mechanisms as contact smart cards and there are several established international standards available to aid interoperability [151].

Contactless systems differ from each other in a number of ways. It is therefore important to consider the alternatives offered by the available standards and products in order to decide on the best system components for a specific application. For example, the three HF standards ISO 14443 [78], ISO 15693 [79] and ISO 18092 [81], allow for different data rates and operational ranges. The growth in the contactless market has also resulted in a variety of readers and tokens. In most cases the reader's only purpose is to act as an RF transceiver between the back-end system, which performs all the processing and security functions, and the token. In general, the only difference between readers are the standards that they support. Readers are also more expensive and are often installed as a long-term infrastructure investment. It is common to find readers supporting multiple standards so that the same hardware can be used in several applications, possibly allowing for future changes and extra functionality.

In contrast, available tokens can by classified in terms of resources, security and interfaces [151]. In terms of resources tokens can be divided into three different types:

- **Memory:** These tokens only have the ability to store information. They can perform no processing in addition to read and write functions.
- **Logic:** In addition to memory, these tokens also include some fixed processing routines that can be invoked by the reader, e.g. authenticate, increment value, decrement value, etc.
- **μ-Controller (MCU):** The token can run custom processing routines and might contain a card operating system such as JCOP or MULTOS.

Tokens can implement a number of security mechanisms. Security increases the token's required resources resulting in a higher system cost, so it is important that the token used provides a level of security sufficient for the application without incurring unnecessary expense. These levels can roughly be defined as follows:

- **Minimal:** Anyone can read information stored on the token. The memory might be locked, so no unauthorized writing of data occurs.
- **Low:** The token implements some form of authentication mechanism. The memory is password protected or mutual authentication must be completed before data is released.
- **Medium:** The token implements a single encryption algorithm used to provide authentication and encryption of data. The algorithm could be proprietary, e.g. NXP Crypto1, or an older industry standard, e.g. DES.
- **High:** The token implements a modern symmetric and asymmetric industry standard algorithms for authentication, encryption, digital signatures, etc.

It may be required that a token supports several technologies. This allows for an environment where the user can carry a single token to access multiple systems. Alternatively, such a token provides a way to migrate to, or add, a new system while still maintaining backward compatibility with existing systems. The following types of tokens have the ability to interact with more than one system:

- **Multiple-Technology:** The token implements multiple technologies. A good example, albeit not contactless, are Chip & Pin cards in the United Kingdom, which contain both magnetic stripe and contact smart card technologies.
- **Dual-Interface:** The token contains one integrated circuit that has more than one interface. An example would be a token containing an IC with both a contactless and a contact interface.
- **Hybrid:** A token with two or more integrated circuits, with their own interfaces, functioning independently. This term can be used to describe HF contactless tokens that also contain additional circuitry to support older Low Frequency (LF) systems.

2.3 Applications

A proximity tokens acts as an electronic credential, interacting with the rest of the system on behalf of the entity it is associated with. Initially these tokens allowed for new applications such as contactless access control and automatic toll collection. In recent years, however, these tokens have started to replace, or supplement, established technologies such as paper tickets for travel or events, barcodes in item tracking and magnetic stripes in credit cards. This section gives an overview of prominent applications in which proximity tokens are used. Table 2.1 summarizes some of these applications.

Application	Standard	Token Resources	Security
Item Tracking	ISO 15693	Memory	Minimal/Low
Ticketing	ISO 15693 ISO 14443	Memory/Logic	Low/Medium
Closed Payment	ISO 14443	Logic	Low/Medium
Open Payment	ISO 14443	μ-Controller	High
Access Control	ISO 14443	μ-Controller	High
Identity	ISO 14443	μ-Controller	High

Table 2.1: Summary of HF RFID applications

2.3.1 Identification

The basic function of RFID, as the name already suggests, is to assist a system in uniquely identifying an item or a person. The simplest example of this is a *tracking* system where a token, storing a Unique Identifier (UID), is attached to an item. The system can then track this item by scanning the token every time it passes a reader. Systems using HF tokens have a shorter operational range than their UHF equivalents although they provide more reliable reader coverage. NXP I-CODE and the Texas Instruments Tag-It products are examples of HF tokens used in tracking applications. In *ticketing* systems, tokens facilitate access to services for a limited time before being disposed of. A ‘ticket’ can take many forms, such as a key card to a hotel room or a day pass at the local gym. Paper tickets, containing NXP Mifare UltraLight tokens, received extensive publicity during the FIFA World Cup in 2006 and were used by spectators to gain access to stadiums, seating areas and refreshment kiosks [133].

Contactless *access control* is popular for securing physical locations. Charles Walton first invented an RFID-based access control system in 1973. The system involved an electronic lock that opened with an RFID key card, which he sold to Schlage [135]. Since then, contactless access control systems have become widespread, not only in the private sector but also with government agencies. An application closely linked to access control is that of *identity*. A token used for proof of identity must contain enough information to allow the system to verify that the person presenting it is the legitimate owner. Identity tokens therefore contain additional personal information, such as biometric data.

The latest US initiative is the Federal Information Processing Standard Publication 201 (FIPS 201), detailing Personal Identity Verification (PIV) of federal employees and contractors [51]. FIPS 201, published by the US National Institute for Standards and Technology (NIST), provides specifications for a standard smart ID card that is to be used for access control. The cards are a requirement for all US federal employees and contractors under the Homeland Security Presidential Directive 12 (HSPD-12). Other large government initiatives in the US include:

- The Department of Defense's Common Access Card with Contactless (CAC-C) being used as identification for on duty military personnel, reserve personnel and civilian employees
- The Transportation Worker Identification Credential (TWIC) being issued by the Transportation Security Administration
- The First Responder Authentication Card (FRAC) being issued in the Department of Homeland Security (DHS) pilots [150]

Proximity tokens are also used for national identity cards and several countries are planning to use contactless ID cards for their citizens, with China [24] recently becoming the largest implementor. The most prominent application of contactless identity is however machine readable travel documents (MRTD). By the 26th of October 2006 the USA required that 27 countries must issue their citizens with e-passports in order to still qualify under the Visa Waiver Program. E-passports adhere to operational specifications as defined by the International Civil Aviation Organisation (ICAO) [75], which is also the standard specified for MRTD in ISO 7501 [76]. By 2015, ICAO wishes to have replaced all current passports with a digital version that stores encrypted biometric data on an RFID chip [8]. The Department of Homeland Security also wants to use passive RFID to record who is entering or leaving the US across land routes, using a People Access Security Service (PASS) card. ICAO allows for optional security protocols that provide both authentication and encryption services. E-passports have the interesting security requirement that anyone who is presented with the passport should be able to read and verify the contents, but at the same time the user's personal details should be afforded some measure of privacy. For this reason most e-passports implement the Basic Access Control (BAC) scheme. BAC derives a key from the passport number, expiry date and the user's birthday, read off the Optical Character Recognition (OCR) strip inside the passport. The idea is that anyone legitimately presented with the passport can read the OCR data, derive the key and retrieve the data off the token inside. The European Union countries are planning to implement Extended Access Control (EAC), involving a public key infrastructure for participating parties, for future passports including additional biometric data [15]. Germany started issuing biometric passports containing fingerprints, protected by EAC, since November 2007 [101].

2.3.2 Payment

RFID has been used in payment systems since the 1980s, when it was first used for automatic toll collection. Since then contactless tokens were implemented in several cashless payment systems [135]. In *closed* systems one organization is in control of the entire payment process. In other words, customers will pay for the organization's services with payment tokens issued by the same organization. These systems often function on a 'prepaid' principle where the customer pays for credit in advance and are ideal for public transport payments. A good example of such a system is the Oystercard scheme implemented by Transport for London (TfL) using NXP Mifare Classic tokens. Closed payment systems can also be used in other environments such as service stations, e.g. the SpeedPass system implemented by ExxonMobil, or in some cafeterias and fast food outlets.

In *open* systems, an organization issues customers with payment tokens that will be used to purchase services from other organizations. Some of these systems still operate using prepaid credit like Hong Kong's Octopus system, implemented with Sony Felica tokens. The Octopus card is used mainly for transport payments, but can also be used to pay at convenience stores, restaurants and other local services. The most prominent open payment system is, however, contactless credit cards. RFID credit cards have been widely deployed in the US [67] with American Express (ExpressPay), MasterCard (PayPass [105]), and Visa (payWave [164]), all supporting contactless credit and debit cards.

In the United Kingdom, the Royal Bank of Scotland, working with Mastercard, and Barclays recently launched contactless debit cards. Barclays' OnePulse card, developed with Visa, is intended to function as a chip-and-pin contact card, contactless debit card and an Oyster card. The lower level communication of these cards, as specified in the EMV Contactless specification [47], adheres to ISO 14443, while the application layer communication adheres to the same Europay, MasterCard, and Visa (EMV) framework and specifications as contact payment cards and transaction terminals [48].

2.4 Radio frequency interface

The operation of contactless systems is based on the principle of inductive coupling. The token and the reader both contain antenna coils that are coupled and interact via a magnetic field. The token receives both data and power from the carrier transmitted by the reader. The token can also send data to the reader by influencing this carrier. Collectively, these methods are referred to as near-field communication since the range between the token is much smaller than one wavelength of the carrier. This section provides an overview of communication theory and physics relevant to contactless systems. The information in this section has been adapted from [54, 129, 130].

2.4.1 Communication theory

In order to transmit information over an RF channel the relevant data must first be encoded and then modulated onto a suitable RF carrier. Line coding changes the binary data into a signal sequence that is best suited to the transmission channel and aids the receiver in recovering the data. Modulation is the process whereby the parameters of an RF carrier are altered in relation to the resultant baseband signal. Modulation, and to a lesser extend coding, techniques can be used to shape the frequency spectrum of the communication channel. In contactless systems, coding and modulation methods have two main prerequisites: To separate the 'weak' backward channel communication from the strong forward channel carrier and to allow for data transfer from the reader to the token while ensuring that the token still receives adequate power from the HF carrier.

A line code's properties will typically be chosen to allow for the physical requirements of the transmission channel. Line codes are most often used to eliminate the DC component

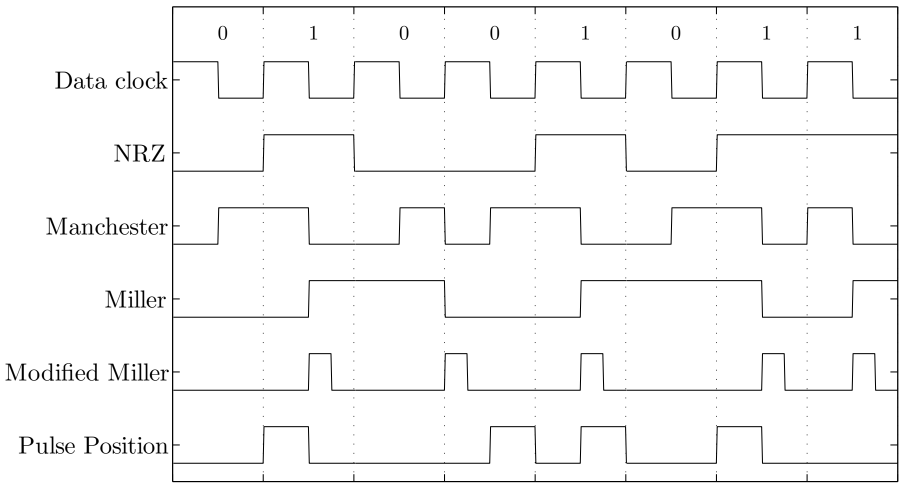


Figure 2.3: Data coding examples

of the data and help the receiver with synchronization, although some codes also provide redundancy to allow errors to be detected and corrected. HF RFID tokens use the schemes shown in Figure 2.3.

Non-Return-to-Zero (NRZ) coding is the most basic coding technique. A ‘1’ is represented by a logical high for one clock period and a ‘0’ is represented by a logical low for one clock period. NRZ encoding is not ideal when used to transmit data without a maximum runlength constraint, or in other words data that contains long sequences of ones or zeros. In this case there will be no signal transitions between high and low for a period of time, which can prevent the recovery of an accurate data clock. This also has other consequences, e.g. if a long sequence of zeros is transmitted using 100 % amplitude modulation it could disrupt the token’s power supply.

Manchester coded sequences have at least one transition during each bit period. This periodic transition, which occurs in the middle of the bit period, can therefore be used to recover the data clock regardless of the data values. A ‘0’ is expressed by a low-to-high transition and a ‘1’ by a high-to-low transition. Manchester encoded sequences contain no DC component, but require approximately twice the channel bandwidth of NRZ.

Modified Miller coding is often used for data transmission from the reader to token. The data is first encoded using Miller coding and then ‘modified’ so that each edge transition, high-to-low and low-to-high, is simply represented by a single pulse. In Miller encoding a ‘1’ is represented by a bit period with a edge transition at the midpoint. A ‘0’, if preceded by a ‘1’, is represented by a bit period with no transition while a ‘0’, preceded by another ‘0’, is represented by a bit period with a transition at the start of the bit period. The advantage of Modified Miller coding is that the carrier is not interrupted for more than the duration of the coding pulse, even if a bit period is very long. This ensures a continuous power supply to the token from the reader’s carrier during data transfer. The bandwidth of Modified Miller coding depends on the duration of the coding pulse.

Pulse-Position coding, which is also sometimes referred to as Pulse-Position Modulation (PPM), represents x data bits with a single pulse in one of 2^x possible time slots. In Figure 2.3 an example of ‘1 in 4’ PPM is given. In this case two bit periods are divided into four time slots. The data bits are then encoded as follows: ‘00’ is represented with a pulse in the last slot, ‘01’ with a pulse in slot three, ‘10’ with a pulse in slot 2 and ‘11’ with a pulse in the first slot. PPM is also suitable for reader to token communication since the carrier is not interrupted for more than the duration of the coding pulse. As with Modified Miller coding the required bandwidth is determined by the time width of the coding pulse.

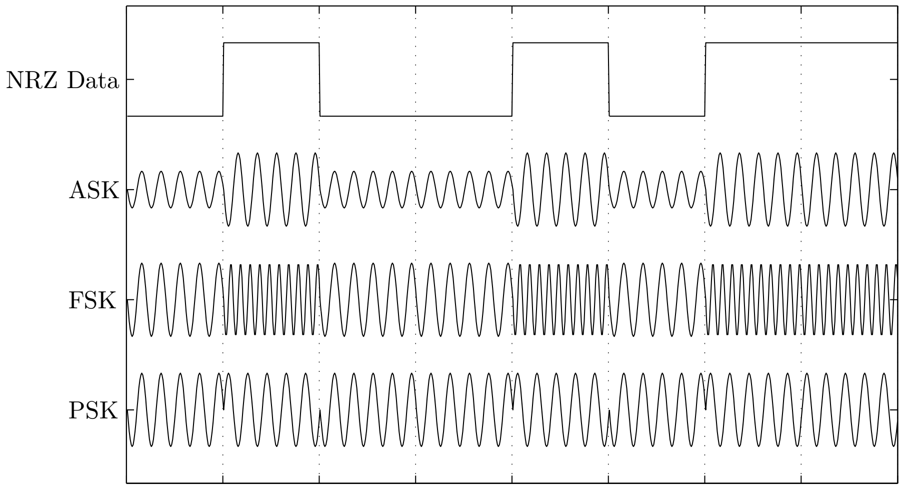


Figure 2.4: Examples of RF modulation schemes

Modulation is the process whereby an RF carrier’s parameters are changed to represent a baseband data sequence. A sinusoidal carrier can be characterized by

$$x(t) = a \cdot \sin\left(2\pi f_{\mathrm{c}} t + \phi\right) \tag{2.1}$$

From the equation it is clear that the amplitude a, frequency f_{c} and phase ϕ can be varied to create distinctive carriers that are suitable to represent different data symbols. The amount that the chosen variable changes in relation to the data is referred to as the modulation index m_i. For example, m_i for amplitude modulation can be represented as

$$\frac{a_{\max} - a_{\min}}{a_{\max} + a_{\min}} \tag{2.2}$$

When modulating digital data, the chosen variable will only change between a set number of discrete values, e.g. a high signal represented by $f = 10$ Hz, a low signal represented by $f = 20$ Hz. As a result modulation is often referred to as ‘shift-keying’ in digital systems. Examples of the modulation schemes used in HF RFID are shown in Figure 2.4.

Amplitude-Shift Keying (ASK) changes the amplitude of the carrier to a level chosen to represent a specific data symbol. In Figure 2.4 a ‘1’ is represented by a carrier with amplitude A and a ‘0’ is represented by a carrier with amplitude $0.5A$. In this case the modulation index would be $0.33 \approx 33\%$. On-Off Keying (OOK) is a special case of ASK where the modulation index is equal to 100%.

Frequency-Shift Keying (FSK) changes the frequency of the carrier to represent a specific data symbol. In the example shown a ‘0’ is represented by a carrier with $f_{\mathrm{c}} = f_1$ and a ‘1’ is represented by a carrier with $f_{\mathrm{c}} = 2 \cdot f_1$.

Phase-Shift Keying (PSK) represents each specific data symbol by a shift in the carrier’s phase. In the example shown a ‘0’ is represented by a carrier with phase equal to 0° and a ‘1’ is represented by a carrier with phase equal to 180°.

The modulation process also changes the frequency domain representation of the data. ASK and PSK cause the power spectrum of the data to move from the baseband to f_c, while the spectrum power components for FSK data, as per our example, will be at f_1 and $2 \cdot f_1$. The amplitude modulation process can be represented mathematically as follows:

$$\begin{aligned} x(t) &= d(t) \cdot \sin(2\pi \cdot f_c t) \\ X(f) &= D(f) * (\delta(-f_c) + \delta(f_c)) \cdot \tfrac{1}{2} \\ X(f) &= (D(f + f_c) + D(f - f_c)) \cdot \tfrac{1}{2} \end{aligned}$$

where $d(t)$ is the baseband data sequence with frequency spectrum $D(f) = \int d(t)e^{-2\pi i f t}dt$ and δ is the Dirac delta function.

This is useful in RFID systems where both the forward and backward channel data are transmitted using the same carrier. The signal power of the backward channel can be up to 80 dB smaller than that of the carrier, which means that the reader would find it difficult to distinguish this 'weak' data from the 'strong' carrier if it cannot effectively isolate the data of interest and attenuate the carrier. Separating the two channels in the frequency domain simplifies the recovery of the backward channel data. In HF RFID the forward channel data is modulated directly onto the main carrier. The backward channel, however, is first modulated onto a sub-carrier before being modulated onto the main carrier. Modulating with a sub-carrier creates two data sidebands separated by f_{sc} from the operational frequency f_c. The sub-carrier modulation process can be represented mathematically as follows:

$$\begin{aligned} x(t) &= (d_B(t) \cdot \sin(2\pi \cdot f_{sc} t)) \cdot \sin(2\pi \cdot f_c t) \\ 2 \cdot X(f) &= (D_B(f) * (\delta(-f_{sc}) + \delta(f_{sc}))) * (\delta(-f_c) + \delta(f_c)) \\ 2 \cdot X(f) &= D_B(f + f_c + f_{sc}) + D_B(f + f_c - f_{sc}) + D_B(f - f_c + f_{sc}) + D_B(f - f_c - f_{sc}) \end{aligned}$$

where $d_B(t)$ is the backward channel data with frequency spectrum $D_B(f)$. The resultant positive frequency spectrum of the forward and backward channels is shown in Figure 2.5. The data at $f_c + f_{sc}$ is referred to as the upper-sideband while the data at $f_c - f_{sc}$ is referred to as the lower-sideband. The backward channel data can now be recovered, despite the presence of a strong operational carrier, by bandpass filtering one of the sidebands. In practice $d_B(t)$ is actually modulated with a square wave carrier, so this mathematical representation is simplified slightly as it shows only the first harmonic of the sub-carrier.

2.4.2 Inductive coupling

HF RFID systems work on the principle of inductive coupling, where one device transfers energy to another by means of a shared magnetic field (H). This operational model is true as long as the token is placed in the near field of the reader, since the high frequency electromagnetic field (E) generated by the reader acts primarily as a magnetic field if the distance between the reader and token is less than $\lambda_{f_c} \cdot \frac{1}{2\pi}$. In simple terms, an RFID system acts in a similar same way to a transformer, with the primary coil contained in the reader and the secondary coil contained in the token. Current flowing through the reader's coil generates a magnetic field, which in turn induces a proportional current flow in the coil of the token. The high frequency carrier can, therefore, be used for both data and power transfer between the reader and the token.

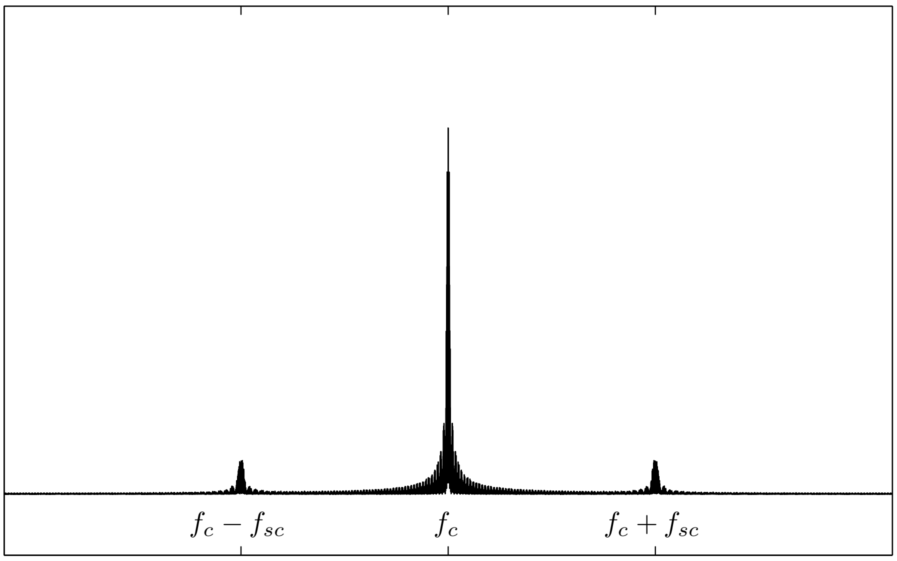


Figure 2.5: Positive frequency spectrum of the forward and backward channel modulated on the carrier with frequency f_{c} and a sub-carrier with frequency f_{sc}

Power transfer

Moving charge, such as the flow of current in a conductor, generates a magnetic field. The magnitude of this field at a specific point is described by the magnetic field strength H. In HF RFID systems, the reader uses conductor loops to generate a magnetic field. The magnitude of the magnetic field generated by these loop antennas depends on the current I, number of loops N, the radius of the loops R and the distance from the loop antenna d. In the near field, the following equation can be used to calculate the field strength along the axis of the loop antenna in the near field [54]:

$$H = \frac{I \cdot N \cdot R^2}{2\sqrt{(R^2 + d^2)^3}} \tag{2.3}$$

In this case d is the distance from the center of the coil along the coil's axis. In general, the field strength is almost uniform at short distances (d < R). For equal I and N, smaller loop antennas can generate a higher field strength although larger antennas have greater field strength at greater distances [161]. A token will specify the minimum field strength it needs to function. Designing the antenna is then a trade-off between making the antenna small enough to generate a strong enough magnetic field and also making it large enough to achieve the system's required operational range.

In an HF RFID system both the reader and the token contain loop antennas in close proximity, as shown in Figure 2.6. If a second loop antenna with area A_2 is located close to another loop antenna with area A_1, then the second antenna will be affected by a proportion of the total magnetic flux Ψ flowing through the first antenna. The two circuits are connected together by this partial transfer of flux and are therefore said to be coupled. The magnitude of the coupled flux Ψ_{21} depends on the characteristics of the loop antennas, the position of the antennas in relation to each other and the magnetic conductivity, or permeability, of the medium between the antennas.

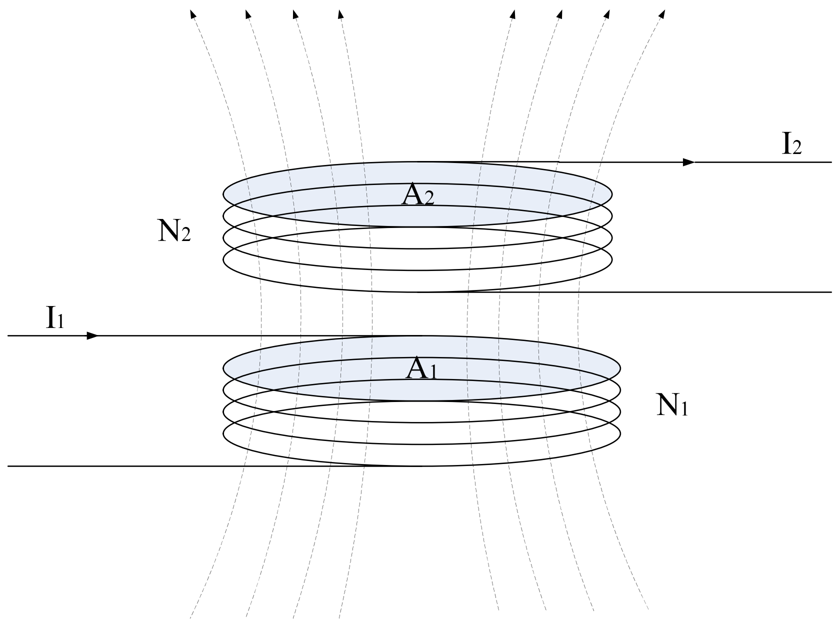


Figure 2.6: Inductive coupling

The ratio of the total flux that is generated in an area, to the current in the conductor that encloses that area, is described by the inductance L [54]:

$$L = \frac{\Psi}{I} = \frac{N \cdot \mu \cdot H \cdot A}{I} \tag{2.4}$$

The constant μ is equal to $\mu_0 \cdot \mu_r$, where μ_0 is the magnetic field constant ($4\pi \times 10^{-6}$) and describes the permeability of a vacuum, while μ_r is the relative permeability, indicating the ratio of the permeability of a material relative to μ_0.

The concept of mutual inductance is used to describe the coupling of two antennas by means of a magnetic field. The mutual inductance M_{21} is defined as the ratio of coupled flux Ψ_{21} enclosed by a second loop antenna to the current I_1 in the first loop [54]:

$$M_{21} = \frac{\Psi_{21}(I_1)}{I_1} \tag{2.5}$$

Similarly there is also a mutual inductance M_{12} although $M_{21} = M_{12} = M$. The mutual inductance between two loop antennas can be calculated using Equations 2.4 and 2.5 and can be approximated as follows [54]:

$$M_{12} = \frac{\mu_0 \cdot H(I_1) \cdot N_2 \cdot A_2}{I_1} \tag{2.6}$$

Replacing $H(I_1)$ with Equation 2.3 and substituting $R^2\pi$ for A the final result is [54]:

$$M_{12} = \frac{\mu_0 \cdot N_1 \cdot R_1^2 \cdot N_2 \cdot R_2^2 \cdot \pi}{2\sqrt{(R_1^2 + d^2)^3}} \tag{2.7}$$

In practice the HF carrier transmitted by the reader is a sinusoidal alternating current. A time varying current $I_1(t)$ flowing in a loop antenna generates a changing magnetic flux. According to Faraday's law a voltage will be induced in the loop antenna that encloses some of this changing flux. Figure 2.7 shows a simplified circuit diagram for a coupled RFID system, where L_1 is the antenna of the reader, L_2 is the antenna of the token, R_{L_2} is the resistance of the token's antenna and R_{LOAD} represents the load. C_{RES} is ignored for now and is equal to 0.

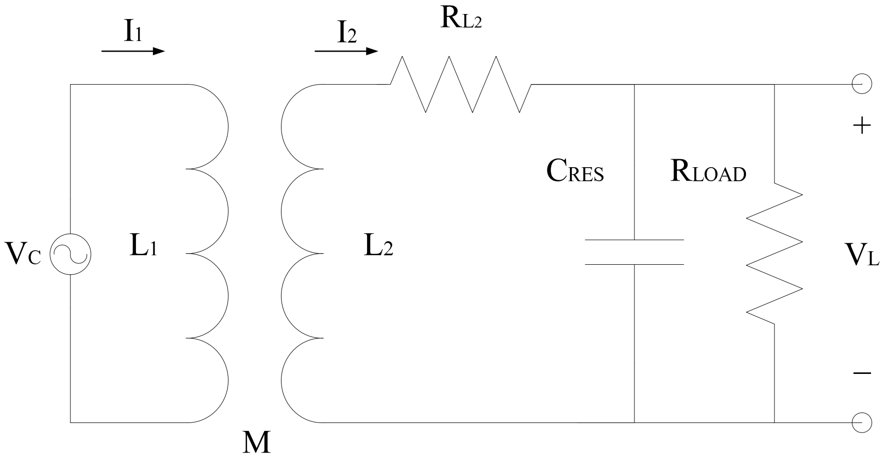


Figure 2.7: Simplified circuit diagram of a coupled token

The total time variant flux in L_1 induces a voltage V_{L1} in L_2 due to the mutual inductance M. There is a voltage drop across R_{L_2} and I_2 also induces magnetic flux in L_2, which opposes $\Psi(I_1)$, so the voltage across the load can be approximated by [54]:

$$V_L = \frac{\mathrm{d}\Psi_2}{\mathrm{d}t} = M\frac{\mathrm{d}I_1}{\mathrm{d}t} - L_2\frac{\mathrm{d}I_2}{\mathrm{d}t} - I_2 R_{L_2} \tag{2.8}$$

V_L can now be rectified and used as a power supply for the token. In order to improve the efficiency of the coupling, an additional capacitor C_{RES} can be added in parallel with the antenna L_2 to form a parallel resonant circuit with a resonant frequency corresponding to the operating frequency of the RFID system. The resonant frequency can be calculated as follows [54]:

$$f_{\mathrm{RES}} = \frac{1}{2\pi\sqrt{L_2 \cdot C_{RES}}} \tag{2.9}$$

When operating at the resonant frequency the voltage V_L induced in a system with a resonant circuit increases by more than a factor of ten compared to a system using the antenna by itself. The influence of the resonant circuit's R_{L_2} and R_{LOAD} on voltage V_L can be characterized by the quality factor Q. The Q factor is discussed in more detail later with regards to data transfer.

A further practical constraint that affects the coupling efficiency is the orientation of the token in relation to the reader. In the equations above it was assumed that the antenna in the reader and the antenna in the token have a common axis. Although this is often the case the token can also be displaced or tilted in such a way that the magnetic flux enclosed by its antenna is decreased. As a rule of thumb maximum voltage is induced in the token's antenna when it is perpendicular to the magnetic field lines, while no voltage is induced if it is parallel. This results in a specific interrogation zone around the reader's antenna, as illustrated by the example in Figure 2.8. Following the expected path of the field lines in this case it can be seen that a token parallel to the reader's antenna would be read if directly in front or to the side of the antenna, while a token that is perpendicular to the antenna could be read on the diagonal corners.

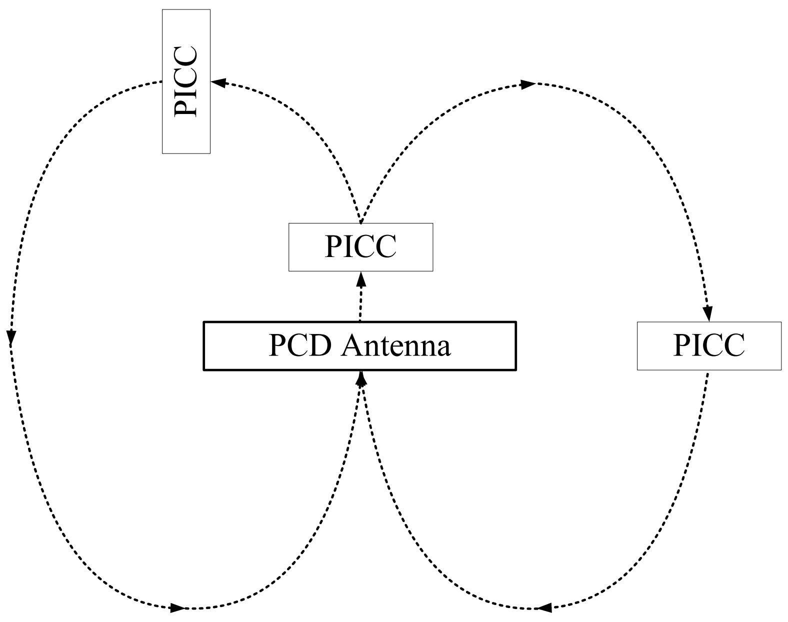


Figure 2.8: Orientation of token to reader antenna for maximum coupling

Data transfer

The reader and token use different techniques to transmit data. As a result reader-to-token communication is referred to as the forward channel, while token-to-reader communication is referred to as the backward channel. The forward channel is relatively simple as the reader can directly modulate the data onto the carrier it transmits. The backward channel, however, requires that the token send data even though it is a passive device. The token must therefore modulate the reader's carrier, which in most cases is done using *load modulation*.

Load modulation works on the principle that the token's impedance Z_T can be altered by changing the parameters of the resonant circuit. Changing the impedance not only influences the voltage induced in the token's antenna L_2 but also the magnitude of the voltage across the reader's antenna L_1. The token can therefore amplitude modulate the voltage on the reader's antenna. It is only possible for the token to alter the load resistance R_{LOAD} or the parallel capacitor C_{RES} of the resonant circuit. Load modulation is therefore either resistive or capacitive. In resistive load modulation a resistor R_{MOD} is added in parallel to R_{LOAD}, so the impedance is switched between $Z_T(R_{LOAD})$ and $Z_T(R_{LOAD}||R_{MOD})$ during the modulation process. In capacitive load modulation an additional capacitor C_{MOD} is added, which changes the resonant frequency when switched into the circuit. Detuning the resonant circuit greatly influences the token's impedance, which causes the desired modulation effect.

As mentioned before the Q factor is a measure of the voltage in the token when operating at the resonant frequency. It can be approximated as follows [54]:

$$Q = \left(\frac{R_{L_2}}{2\pi f_c L_2} + \frac{2\pi f_c L_2}{R_{LOAD}} \right)^{-1} \tag{2.10}$$

Generally the value of Q should be maximised to allow for the maximum power transfer and operational range. It should be kept in mind that Q also influences the bandwidth

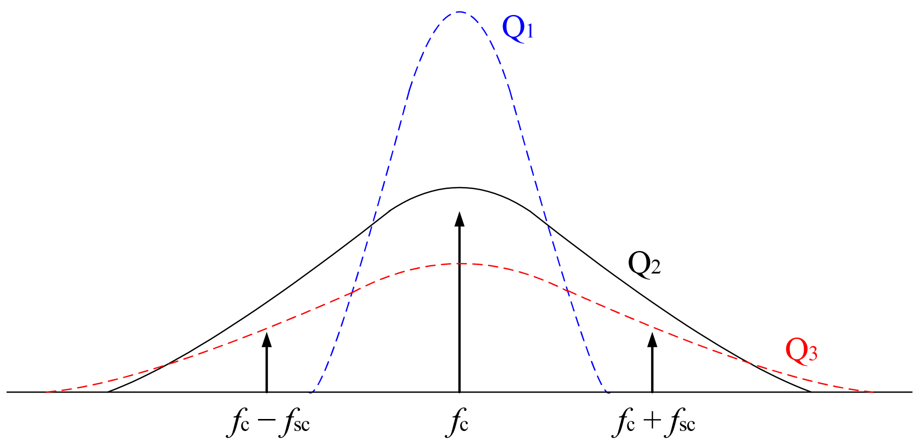


Figure 2.9: The effect of the Q-factor

of the communication channel. The frequency response of the system peaks around the resonant frequency and rolls off to either side, as shown by the examples in Figure 2.9. This indicates that the resonant circuit acts as a crude bandpass filter centered around f_c with bandwidth $BW = f_c/Q$. The value of Q must therefore be chosen in such a way that it allows sufficient power transfer, while still allowing for the backward channel's modulation sidebands. In Figure 2.9 the value of Q_1 is too high since it excludes the modulation side bands, whereas the value of Q_3 is too low since it attenuates the RF carrier, of frequency f_c, thereby decreasing power transfer to the token.

2.5 Standards

RFID technology encompasses a range of systems from multiple vendors. To ensure interoperability between different RFID systems several standards have been defined to which these systems must adhere. In the HF band there are three main standards by ISO/IEC (International Organization for Standardization / International Electrotechnical Commission) that deal with HF RFID technology. These are ISO/IEC 14443 [78], 'proximity' tokens with operating range less than 10 cm, ISO/IEC 15693 [79], 'vicinity' tokens with operating range less than 1 m, and finally ISO/IEC 18092 [81], which specifies near-field communication for active devices such as mobile phones. Another standard that should be mentioned is ISO/IEC 18000, which defines alternative RFID communication interfaces for several operating frequencies, including 13.56 MHz, with specific emphasis on tags used for automatic identification and data capture within supply chain applications. The standards define, amongst other things, the RF interface, the initialization sequence and the data format. It is not feasible to describe each standard in its entirety within this chapter, so I only provide a summary of the key technical aspects of the interaction between the reader and the token. ISO 14443 and ISO 15693 are discussed in more detail since these are the most relevant to the applications described in Section 2.3, with ISO 18000 and ISO 18092 discussed only briefly. The information in this section has been adapted from [54, 78–81] and the reader should consult these sources for a more complete description.

2.5.1 ISO 14443

ISO 14443, titled *Identification Cards – Proximity Integrated Circuit Cards*, is commonly used in systems using logic and μ-controller tokens. This means that it is the standard of choice for e-passports, credit cards and most access control systems. The popular Mifare range of products by NXP also adhere to Part 1–3 of ISO 14443 Type A [122].

Part 1 – Physical characteristics

The first part of the standard states that the token shall have physical characteristics according to the requirements for the card type ID-1 specified in ISO/IEC 7810, i.e. 85.72 mm $\times$ 54.03 mm $\times$ 0.76 mm. It also specifies tolerance levels for the token with regard to ultra-violet light, X-rays, dynamic bending stress, dynamic torsional stress, alternating magnetic fields, alternating electric fields, static electricity, static magnetic fields and operating temperature.

Part 2 – Radio frequency power and signal interface

The second part of the standard describes the signal characteristics of two types of radio interfaces between the token and the reader. These interfaces allow for both power transfer and bi-directional communication. The token's power is provided by an alternating magnetic field at a frequency of 13.56 MHz and the reader must ensure that the magnetic field is within the range $1.5\ \text{A/m} \leq H \leq 7.5\ \text{A/m}$. The operational range for systems using ISO 14443 usually extends up to 10 cm.

ISO 14443 defines two different methods for data transfer. In *Type A* the forward channel data uses Modified Miller coding with a coding pulse of 2–3 µs modulated onto the 13.56 MHz carrier with 100 % ASK. The backward channel uses Manchester encoding, which is 100% ASK modulated onto a 847 kHz sub-carrier before being load modulated onto the 13.56 MHz carrier. In *Type B* the forward channel uses NRZ encoding modulated onto the 13.56 MHz carrier with 10 % ASK. The backward channel uses NRZ encoding, which is first modulated onto a 847 kHz sub-carrier using PSK (0°, 180°) before being load modulated onto the 13.56 MHz carrier. The basic data rate for both the forward and backward channels in Type A and Type B is 106 kbit/s. Some ISO 14443 tokens and readers support higher data rates, such as 212, 424 or 848 kbit/s, which can be selected after the anti-collision process has finished.

Part 3 – Initialization and anti-collision

This part of the standard describes byte and data frame formats and the initial commands used to detect and initialise communication with the token. This includes the ability to poll for new tokens in the interrogation field and choosing a token, even if multiple tokens are present, through an anti-collision process.

Type A: A data frame is identified by Start-of-Frame (SoF) and End-of-Frame (EoF) symbols and may contain multiple bytes. Each byte of data is followed by an odd-parity bit. Each frame also contains a 2-byte Cyclic Redundancy Check (CRC) that is appended at the end of the data. During the initialization phase the token acts like a state machine.

The reader then issues commands to change the state of the token as required. A simplified Type A state machine is shown in Figure 2.10. The reader uses the following commands:

- *REQA/WUPA*: The Type A Request (*REQA*) and the Type A Wake-Up (*WUPA*) commands are periodically sent by the reader to poll its interrogation field for tokens. The *WUPA* command can also be used to put tokens that have entered the HALT state into the READY state.
- *SELECT (NVB < 40)*: If the Number of Valid Bits (NVB) is less than the number of bits in the Unique Identifier (UID) then the *SELECT* command is used for anti-collision, thus it is also referred to as the *ANTICOLLISION* command.
- *SELECT (NVB = 40)*: When the reader has determined the UID of the token it wishes to communicate with, it uses the *SELECT* command to place that token in the ACTIVE state.
- *HLTA*: The Type A Halt (*HLTA*) puts the token into the HALT state.

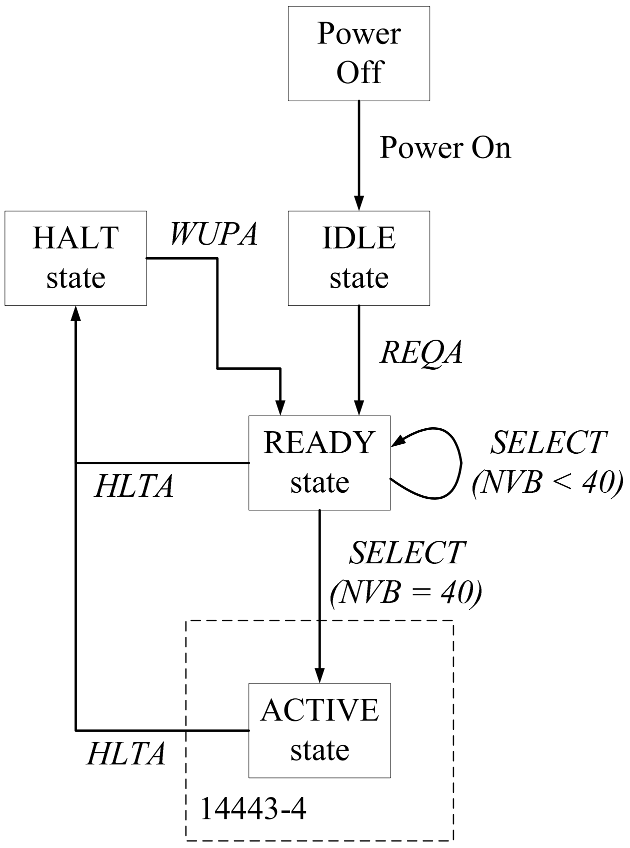


Figure 2.10: Type A: Token State Machine

When the token enters the interrogation zone of the reader it powers up and enters the IDLE state. In this state the token will not respond to any commands from the reader except *REQA* and *WUPA*, which will put it into the READY state. Readers will periodically transmit a *REQA* command to see if there are tokens within its interrogation zone. If it receives a Type A Answer To Request (ATQA) response it knows that a token is present and will proceed to the next step of initialisation. It is often the case that multiple tokens will be presented to the reader at once, e.g. travel and credit cards in the same wallet, so the standard must allow the reader to select a specific token. This process of selection is known as anti-collision, which in Type A is implemented using a binary search tree algorithm. The *SELECT* command is a bit-oriented frame containing a field indicating the length of the current search (NVB) and a search pattern. If the least significant bits of a token's unique identifier match the search pattern for the specified search length, that token will respond with the rest of its unique identifier. An example of the anti-collision process is shown in Figure 2.11. Please note that NVB has an offset of 32, which is 20h when represented in hexadecimal. In the first step the reader sets the search length to 0 (NVB$= 20h+0h$), which results in both tokens transmitting their whole 4-byte unique identifier and a checksum (BCC). The first collision that the reader detects is in the fifth bit. The reader therefore sets the search length to 5 (NVB$= 20h+5h = 25h$) and transmits a search pattern that indicates to the token whether the fifth bit should be a '1' or a '0'. Only the first token's unique identifier matches the search string, so it alone responds with the rest of its identifier. Since the reader detects no collisions it knows that it has identified a single token. Finally the reader sends the same *SELECT* command with maximum search length and full unique identifier to which the tokens responds with a Select AcKnowledge (SAK). This also results in the token being put in the ACTIVE state.

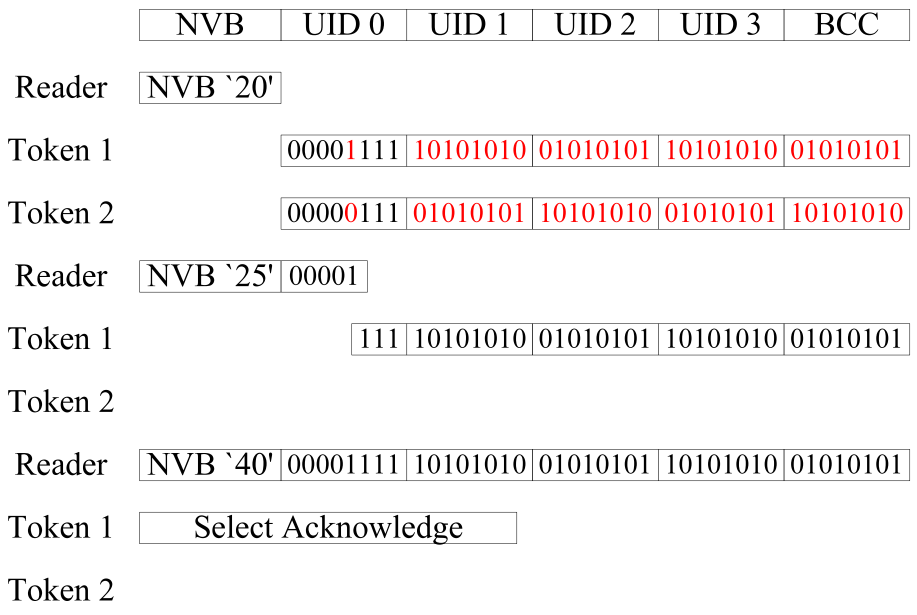


Figure 2.11: Type A: Example of an anti-collision sequence

For the anti-collision to work, the token's responses must be closely synchronized with the reader's commands. Type A expects the token to behave in a synchronous manner, so it prescribes a fixed bit grid which defines when a response must be sent. This grid is defined by specifying a Frame Delay Time (FDT) as follows: FDT $= (n \cdot 128 + 84)/f_c$ if the last bit sent by the reader is a '1' and FDT $= (n \cdot 128 + 20)/f_c$ if the last bit sent by the reader is '0'. n is equal to 9 for *REQA*, *WUPA* and *SELECT* commands, while $n \geq 9$ for all other commands.

Type B: A data frame, marked with start-of-frame and end-of-frame symbols, contains multiple characters. Each character consists of a start bit, eight data bits and a stop bit, which must be followed by a set Extra Guard Time (EGT) before the next character starts. Each frame also contains a 2-byte cyclic redundancy check that is appended at the end of the data.

As with Type A the token acts like a state machine during initialization. A simplified Type B state machine is shown in Figure 2.12. The reader uses the following commands:

- *REQB/WUPB*: The Type B Request (*REQB*) and the Type B Wake-Up (*WUPB*) commands are periodically sent by the reader to poll its interrogation field for tokens and initiate the anti-collision procedure. The *WUPB* command can also be used to put tokens that have entered the HALT state into the IDLE state. *REQB* and *WUPB* commands contain an Application Family Identifier (AFI) that indicates the type of application targeted by the reader. This field is used to preselect tokens participating in the anti-collision process since only tokens with an application of the type indicated by the AFI may answer with a Type B Answer To Request (*ATQB*).
- *SLOT-MARKER*: During the anti-collision process the reader may send up to $N-1$ *SLOT-MARKER* commands to indicate each time slot that is available for a token's *ATQB* response. The commands can be sent after the end of a received *ATQB* message to mark the start of the next slot or earlier if no *ATQB* is received and it is known that the slot will be empty.
- *ATTRIB*: The *ATTRIB* command is used by the reader to select a single token. Upon receiving an *ATTRIB* command containing its identifier, a token enters the ACTIVE state where it only responds to commands, as defined in ISO/IEC 14443-4, that include the Card Identifier (CID) assigned to it in the *ATTRIB* command parameters.
- *HLTB*: The Type B Halt (*HLTB*) puts the token into the HALT state.

When the token enters the interrogation zone of the reader it powers up and enters the IDLE state. The anti-collision procedure, based on a dynamic slotted ALOHA algorithm, is started when the *REQB* command is transmitted. The token checks to see if it has an application that matches the received AFI parameter and if this is the case it calculates the number of anti-collision slots N from the parameters in the *REQB* command. If $N = 1$ the token responds immediately with its *ATQB* response. Alternatively the token is put in the READY REQUESTED state and randomly calculates a slot in which to send its response. In a probabilistic system, which does not use time slots, a token responds only if its randomly chosen slot is equal to 1. If it chooses any other slot it returns to the IDLE state and waits for the anti-collision procedure to start again. In a pseudo-deterministic system that uses multiple slots, the tokens responds within the time slot corresponding to its randomly chosen slot. Once a token has responded with an *ATQB* it is in the READY DECLARED state. The *ATQB* response contains information that the reader can use to identify and select the token. Once the reader has received a collision-free *ATQB* from the card it wants to select, it uses the *ATTRIB* command to place the chosen token into the ACTIVE state.

Part 4 – Transmission protocol

The final part of the standard specifies a half-duplex block transmission protocol for communication between the reader and the token. In Type A tokens, additional setup

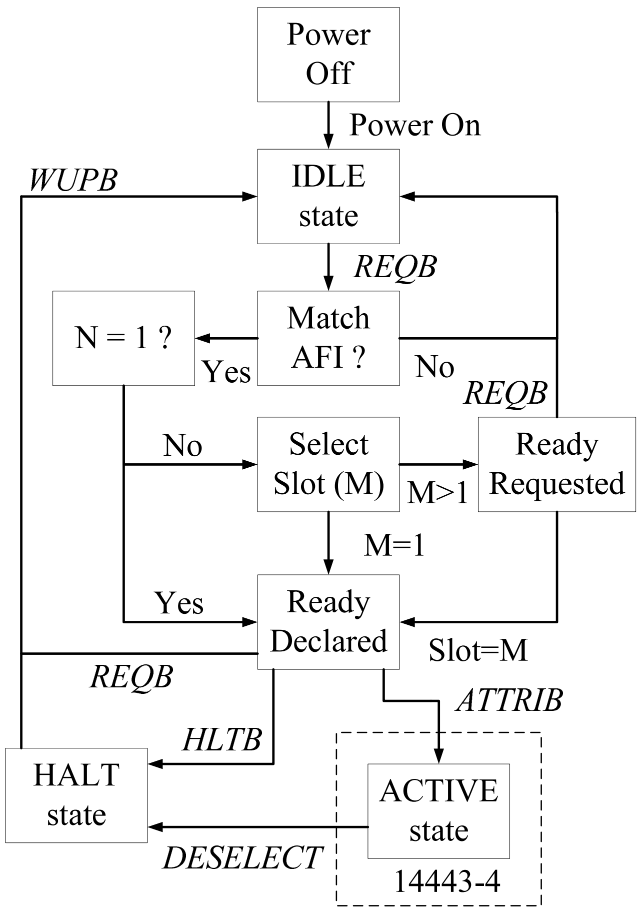


Figure 2.12: Type B: Token State Machine

parameters need to be exchanged between the token and the reader to configure the protocol, such as frame size, card identifier, etc. In Type B tokens this is not necessary as these parameters have already been exchanged in the $ATQB$ response and the $ATTRIB$ command. When the Type A token is selected, the select acknowledgment will contain information about whether the token implements a proprietary protocol, or whether it supports ISO 14443-4. If a protocol adhering to ISO 14443-4 is available, the reader will transmit a Request for Answer To Select ($RATS$) command to which the token replies with an Answer-To-Select (ATS). This is followed by Protocol and Parameter Selection (PPS), if supported, that allows for the change of data rate between the token and the reader. The transmission protocol itself allows for the transmission of an Application Protocol Data Unit (APDU) that can contain any required data. The structure of the protocol is based on the $T = 1$ protocol specified in ISO 7816-3 for contact cards, and is therefore often referred to as $T = CL$. This simplifies integration of contactless applications into smart card operating systems that are already available, especially in dual interface tokens.

2.5.2 ISO 15693

ISO 15693, titled *Identification Cards – Contactless Integrated Circuit Cards – Vicinity Cards*, is most often implemented in systems using memory tokens for tracking or simple identification. Part 1 of the standard is very similar to the corresponding part in ISO 14443, so I discuss only Parts 2 and 3.

Part 2 – Air interface and initialization

This part of the standard describes the signal characteristics of the radio interface between the token and the reader, which allows for both power transfer and bi-directional communication. The token's power is provided by an alternating magnetic field at a frequency of 13.56 MHz and the reader must ensure that the magnetic field is within the range 115 mA/m $\leq H \leq$ 7.5 A/m. The operational range for systems using ISO 15693 can extend up to 1 m.

ISO 15693-2 defines both 'long distance' and 'fast' communication modes. In the 'fast' mode the forward channel uses 1 of 4 pulse-position coding that is modulated onto the 13.56 MHz carrier with 100 % ASK. One symbol comprises 8 time slots each of duration 9.44 μs and the modulation pulse can only be transmitted at an odd-numbered time slot. The value n of the symbol can be determined by pulse slot $= ((2 \cdot n) + 1)$ and can be 0, 1, 2 or 3. One symbol takes $8 \times 9.44\ \mu$ s = $75.53\ \mu$s to transmit, but each symbol can convey two bits of data, so the data rate is 26.48 kbit/s. For the 'long distance' forward channel data, 1 of 256 pulse position coding is used, which is then 10% ASK modulated onto the 13.56 MHz carrier. One symbol now comprises of 512 time slots and takes 4.833 ms to transmit. Since the symbol can have any value between 0 and 255 it can represent 8 bits of data so the effective data rate is 1.65 kbit/s. The backward channel uses Manchester coding, which can either be ASK modulated onto a 423 kHz sub-carrier or FSK modulated with a 423/485 kHz sub-carrier before being load modulated onto the 13.56 MHz carrier. For 'long distance' mode the data rate is 6.62 kbit/s and for 'fast' mode the data rate is 26.48 kbit/s. Data are transmitted in frames marked by a start-of-frame and end-of-frame symbol.

Part 3 – Anti-collision and transmission protocol

The final part of the standard provides details on the anti-collision procedure, token initialization and possible commands. It should be noted that a large section of the guidelines given in this part is optional. As in ISO 14443 the token acts like a state machine during initialization. A possible state machine diagram, given as an example in the standard, is shown in Figure 2.13. The standard's transmission protocol specifies the format for command requests and responses, including a format for a number of optional commands such as *SELECT*, *RESET*, *READ* and *WRITE*. The standard only specifies two mandatory commands:

- *INVENTORY*: The token participates in the anti-collision process when receiving the *INVENTORY* request.
- *STAY-QUIET*: If the token receives the *STAY-QUIET* request it shall enter the QUIET state.

When the token enters the interrogation zone of the reader it enters into the READY state. The reader will poll for tokens by transmitting an *INVENTORY* request. If a token is present it will participate in the anti-collision procedure. The token compares the mask value from the *INVENTORY* request, varying in length from 0 to 8 bytes, with the corresponding least significant bits in its 64-bit unique identifier. If it is a match, the token will reply with its identifier during one of sixteen slots marked by the reader. It is

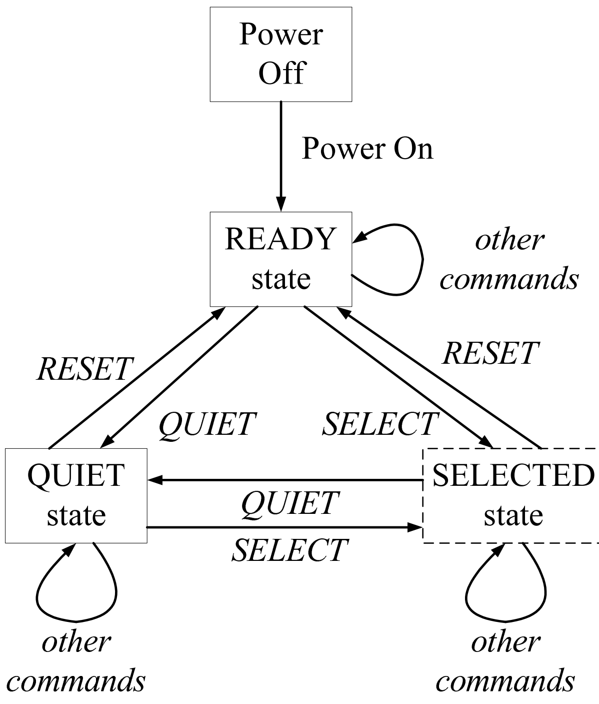


Figure 2.13: ISO 15693: Possible token state machine

assumed that the token does not have the necessary resources to randomly choose a slot. It therefore uses the four least significant bits, not compared with the mask, to determine the slot number. For example if the 2 least significant bytes of the tokens UID is *4FAC* and the mask was *FAC* then the token will respond in slot number 4. Once the reader has the token's unique ID it can put the token in the QUIET state, or in the SELECTED state, if supported, where further commands can be issued.

2.5.3 ISO 18000

ISO 18000, titled *Information Technology AIDC Techniques – RFID for Item Management – Air Interface*, defines a generic structure for use in item management applications (Part 1), along with air interfaces for operation at 135 kHz (Part 2), 13.56 MHz (Part 3), 2.45 GHz (Part 4), 5.8 GHz (Part 5), 860–930 MHz (Part 6) and 433 MHz (Part 7). It is expected that all the parts of ISO 18000 will still be revised to include fixes and allow for the extra capabilities, such as active tokens and sensors [74]. This section only gives a brief overview of part 3 of the standard in comparison to Part-2 of the ISO 14443 and ISO 15693 standards.

Part 3 – Parameters for air interface communications at 13.56 MHz

ISO 18000-3 defines a physical layer, collision management system and protocol values, in accordance with ISO 18000-1, for RFID systems operating at 13.56 MHz. It specifies two modes of operation intended for use with different applications. Mode 1 is based on ISO 15693 with additional commands to allow for item management applications and improved vendor compatibility. The reader to token data rate is 1.65 kbit/s, or 26.48 kbit/s. The token to reader data rate is 26.48 kbit/s. A protocol extension allows for data rates of 53 and 106 kbit/s. Mode 2 specifies a high speed interface where the reader to tag data rate is 423.75 kbit/s.

2.5.4 ISO 18092/NFC

ISO 18092, which is also referred to as NFCIP-1 (*Near-Field Communication Interface and Protocol*) or ECMA 340, specifies an RF interface and transmission protocol for communication between two inductively coupled devices operating at a frequency of 13.56 MHz. This standard allows for an active device, such as a mobile phone or PDA, to access RFID applications by acting as a reader, or a passive token, and it can also be used for short range peer-to-peer communication. Although relatively new, NFCIP has strong support from industry. An NFCIP device can operate in three different ways and it has more resources than a passive token, thereby allowing it to interface with its environment in a number of ways. The standard itself and the additional specifications for data exchange formats, record types and compatible tokens are therefore quite comprehensive. I will only briefly discuss the different modes of operation and the sections of the standard corresponding to Part-2 of the ISO 14443 and the ISO 15693 standards. The reader is encouraged to read the ECMA 340 documentation [46] or visit the NFC Forum [117] for more details.

This standard defines ‘active’ and ‘passive’ communication modes between two entities, referred to as the target and the initiator. In active mode both the initiator and the target generate an RF field. The initiator will start communication and transmit data by modulating its own carrier. Once it has finished transmitting it will switch off the carrier and wait for a response. This is similar to two readers transmitting data to one another. The target then switches on its carrier and transmits a response. In the passive communication mode only the initiator generates an RF field and starts the communication by modulating data on its own carrier. The target then responds to the initiator using a load modulation scheme. In this mode one device acts like a reader while the other device emulates a passive token. Each device must ensure that it generates a magnetic field at a frequency of 13.56 MHz that is within the range $1.5 \text{ A/m} \leq H \leq 7.5 \text{ A/m}$. The operational range for NFCIP systems is in the order of a few centimeters.

Both active and passive modes are defined for communication rates of 106, 212 and 424 kbit/s. The method for transmitting data at 106 kbit/s in passive mode is the same as for 14443A. 106 kbit/s data transmission in active mode uses the same modulation scheme as the forward channel in 14443A. For 212 and 424 kbit/s ‘passive’ and ‘active’ modes both the forward and backward channel uses Manchester code that is ASK modulated onto the 13.56 MHz carrier with a modulation index of 8% to 30% [81]. It should be noted that the backward channel does not use sub-carrier modulation. NFCIP technology was initially developed by Nokia, NXP (previously Philips Semiconductors) and Sony, which is probably the reason why the lower layer communication is compatible with Mifare (106 kbit/s) and FeliCa (212/424 kbit/s) products.

Chapter 3

Security of near-field channels

RFID systems often use near-field magnetic coupling to implement communication channels. The advertised operational range of these channels is less than 10 cm and therefore several implemented systems assume that the communication channel is location limited and therefore secure. Nevertheless, there have been questions asked about whether near-field systems are vulnerable to eavesdropping and skimming attacks. In this chapter I discuss these attacks and present my results from practically implementing them against near-field systems. I also discuss the possibility of using additional cover noise as a countermeasure against eavesdropping.

3.1 Introduction

RFID tokens, using near-field channels, are used to store valuable information in cashless payment systems and even travel documents. No physical contact needs to be made with the reader, which simplifies operation and increases overall transaction speeds. This lack of human interaction has however led to fears that this technology could be abused and as a result most RFID security discussions have centered around privacy concerns. Some consumer groups claim that information about the user could be acquired without consent, and therefore they have rallied against the ‘big brother’ potential of RFID technology [142]. As RFID tokens are used for transactions of increasing value, they could also become the target of attackers, who, if able to gain access to the information on the RFID token, might be able to engage in the act of ‘digital pick-pocketing’ while just standing next to the victim.

The two main attacks to consider are skimming and eavesdropping. When I started working on this topic there were already several claims about the possibility of these attacks on RFID tokens. There were, however, no published scientific work describing practical implementation even though there were rumours regarding successful attack experiments, e.g. e-passport eavesdropping [173]. My main contribution was to practically demonstrate eavesdropping and skimming attacks at a time when there was much debate about the possibility of such attacks succeeding against tokens using near-field communication, i.e. it was believed to be possible but still had to be implemented to be considered a real threat to deployed systems. The practical feasibility of an attacker implementing these attacks, the cost and effort required by the attacker and the distances at which these attacks are possible are often debated and used as an indication of RFID security, for example [127], so this is an important factor when considering the threat model for RFID devices. Implementing proof-of-concept attacks established a benchmark for attack distances and also allowed me to document my experimental setup and method. The RF equipment I used were previously purchased or based on publicly available design, the data recovery method is already known and the external factors mentioned, e.g. environmental noise, should be familiar to most RF engineers. It was not my intention to discover new methods of attack but to explain what an attacker would need to do and provide a good indication of the feasibility of these attacks, especially with regards to the skill and resources required by an attacker. I hope that my work would allow system designers to comprehend the eavesdropping and skimming threat in order to select appropriate technologies and countermeasures, and provide further information for researchers and end users.

In this chapter I discuss the implementation of eavesdropping and skimming attacks on HF RFID systems using near-field communication. I present results for eavesdropping on RFID systems using the ISO 14443A/B and ISO 15693 standards. I also present results, and an alternative implementation, for skimming RFID tokens. In each case I provide a detailed explanation of the attack and description of the experimental setup. This chapter is partly based on a short paper that I authored in 2005: “Practical Attacks on Proximity Cards” [61]. This was the first academic paper, to my knowledge, describing practical results for these attacks in the context of HF RFID systems. At the end of the chapter, I discuss the possibility of creating an eavesdropping resistant communication channel using bit-blocking and random cover noise. This is based on two publications I authored in 2007, “Noisy Carrier Modulation for HF RFID” [60] and “Modulating a Noisy Carrier Signal for Eavesdropping-Resistant HF RFID” [59].

3.2 Attack scenarios

A growing security concern with RFID devices is the possible release of the user’s personal information, or location, to unauthorized parties. The two obvious ways in which information can be released to an unauthorized party are skimming and eavesdropping, as shown in Figure 3.1. In most cases the attacker can execute these attacks from further than the operational range, which is the distance at which the system is expected to communicate. In each attack case there are different scenarios, resulting in different attack distances [83].

Skimming occurs when the attacker uses his reader to access information on the victim’s RFID token without consent. The attacker has the ability to read stored information or to modify information by writing to the token, so he can control when and where the attack is performed. In practice the attacker’s main challenge is to increase the operational range by powering and communicating with the token over a greater distance, as the owner might become suspicious of somebody in his personal space. In this attack there are two distances to consider:

- The distance at which an attacker can power the token and issue a command.
- The distance at which an attacker can power the token, issue a command and recover a response.

Eavesdropping occurs when the attacker intercepts communication between an RFID token and an authorized reader. The attacker does not need to power or communicate

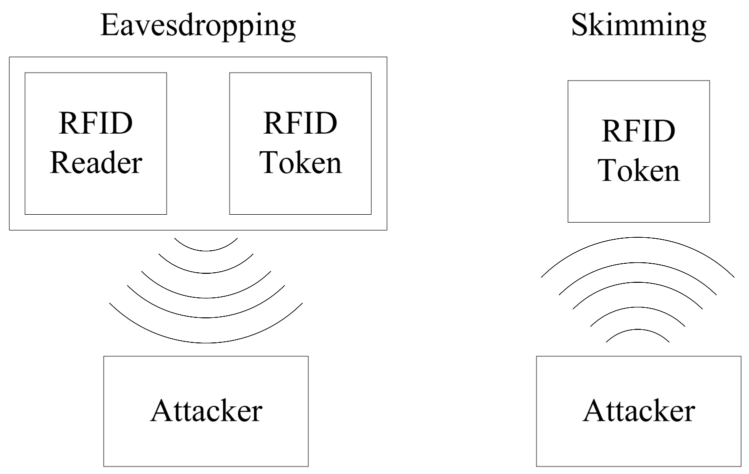


Figure 3.1: Eavesdropping and skimming attacks

with the token, so he is able to execute the attack from a greater distance than is possible for skimming. He is, however, limited in terms of location and time window, since he has to be in the vicinity of an authorized reader when a transaction that he is interested in, is conducted. In the HF RFID standards the communication schemes used for reader-to-token (forward channel) and token-to-reader (backward channel) are different. As a result the distances at which an attacker can recover the data sent on the forward and backward channels differ. There are three distances to consider for this attack:

- The distance at which an attacker can detect a transaction, i.e. he can see activity on the forward channel but cannot reliably recover the actual data.
- The distance at which an attacker can reliably recover the data sent on the forward channel.
- The distance at which an attacker can reliably recover the data sent on the backward channel.

I assume that an eavesdropping attack is successful when the attacker can reliably recover both the forward and backward channels. Eavesdropping and skimming attacks are described in more detail in Sections 3.3 and 3.4.

3.2.1 Related work

Eavesdropping and skimming attacks are not new and are mentioned regularly in the literature. Recent reports by the National Institute of Standards and Technology (NIST) [118], the Department of Homeland Security (DHS) [44] and the German Federal Office for Information Security (BSI) [16], along with academic surveys, e.g. [83], all mention scenarios for eavesdropping and skimming attacks in the RFID environment. These reports, however, do not show practical results or fail to clarify the experimental setup if they do.

Different scenarios exist for eavesdropping and skimming attacks and therefore the experimental setup should be known in order for published results to be useful. In earlier reports terms used to describe the attacks were also confusing. A report on 'Port of Entry' tests done in 2004 [49] states that signals from e-passport systems could be 'detected' at 20 m. The report does not explain whether this implies that the attacker could detect that a transaction occurred, or whether he could recover the actual data. The test also covered a number of different systems and no details were given about which system yielded the result. There were also press reports that NIST eavesdropped the RFIDs to be used in USA passports from as far away as 9 m [173]. Reports, however, often used the term 'read', which implied a skimming attack, while they were actually describing eavesdropping. There are also cases where reports do not state clearly which type of token they were referring to when describing attack distances. RFID is a collective term for several systems and in reality refers to devices adhering to a number of different standards. An HF token used for a contactless smart card is not the same as a UHF token used in logistics. Therefore, if somebody can read a razor's tag from 1 m it cannot be assumed that the same is true for an e-passport. It is therefore important to clearly state the type of RFID system when describing these attacks. Yet the American Civil Liberties Union (ACLU) demo, where a 'passport' was read from 1 m, used 'similar' RFID technology and not an ISO 14443 token as used in a real e-passport [127].

I started my work on implementing practical attacks in 2005. Shortly thereafter, a work-in-progress report was released by researchers at the BSI, where they demonstrated eavesdropping at a distance of 2 m [53] on an ISO 14443A card. Riscure, a Dutch security company, later claimed that it was possible to eavesdrop the backward communication at a distance of 5 m, and the forward channel at a distance of 25 m [143]. They have, however, not actually implemented the attack. In 2006 Kirschenbaum and Wool demonstrated and documented a skimming attack on ISO 14443A tokens at a distance of 25 cm [92]. They used a loop antenna with a diameter of 40 cm to power and read a card. In the same year Flexilis demonstrated a skimming attack for 21 m on a UHF [95] system. At the end of 2006 NIST published a report [58], which was reported in [118], to show that ISO 14443 tokens could be eavesdropped at 15 m. I first obtained a copy of this document in January 2008, the content of which is discussed together with my own results in Section 3.3.3.

Several other research papers describe practical projects regarding HF RFID, which require that the system receives RF communication, although none of these can be said to have implemented an eavesdropping attack [25, 141]. A number of published security protocols make the assumption that the data transmitted from the token to the reader is secure, or more so than the data transmitted by the reader to the card [168].

3.2.2 Significance

The recovery of useful data by eavesdropping can be prevented by encrypting the transmitted data, and skimming attacks can be prevented by implementing suitable authentication mechanisms. Most HF RFID tokens are basically contactless smart cards, which can easily cope with implementing application layer security. So why are these attacks still important? In earlier systems near-field communication was seen as secure because the specified operational range was seen to be limited and as a result several weak security measures were implemented. This section briefly discusses two security sensitive RFID applications and their perceived weaknesses soon after deployment.

Credit cards: In the United Kingdom several contactless payment systems were launched recently, as discussed in Chapter 2. The majority of these systems adhere to ISO 14443A. Plans have also been put forward where an ISO 18092 enabled device, such as a mobile phone or PDA, acts as a contactless credit card [120]. RFID credit cards have, however, been used in the USA since 2003 [167], where these were also implemented using the

ISO 14443B communication standard. Not enough information is currently available to comment on the new contactless payment systems in the UK, but a study has shown there to be a number of vulnerabilities in the first generation of USA credit cards. User and banking information were often sent in plaintext between the reader and the RFID-enabled cards. An attacker could also retrieve the data by implementing a skimming attack and it was proposed that the information transmitted on the RF channel was sufficient to imitate a valid card [67].

e-Passports: By 26 October 2006 the USA required that 27 countries issue their citizens with e-passports in order to still qualify under the Visa Waiver Program. E-passports adhere to operational specifications as defined by the International Civil Aviation Organisation (ICAO) [75] and use the ISO 14443 standard. ICAO allows for optional security protocols, such as Basic Access Control (BAC), that provides both authentication and encryption services. BAC derives a key from the passport serial number, expiry date and the user's birthday, read off the OCR strip inside the passport. The idea is that anyone presented with the passport can read the OCR data, derive the key and retrieve the data off the RFID token inside. Security problems of this scheme have been pointed out [84], especially with the effective size of the key. Theoretically the data can be used to generate a key with an effective length of at least 50 bits [97]. Predictability in the data could however decrease the effective key length to 35 [143] or even 27 [84] bits, which makes a brute force key search attack feasible. This implies that an attacker could eavesdrop communication between a passport and reader and try to decrypt it at a later stage exploiting this weakness in the key. In this case skimming attacks are more difficult, since the attacker would need access to the passport until he finds the correct key by repeatedly running the authentication protocol. It might, however, be feasible that an attacker can gain access to the passport while in the mail and attempt to read the data without breaking the seal on the envelope, especially when the attacker already knows some of the information used to generate the key [138]. Some passports contain shielding, which works on the principle of a Faraday cage. Although this approach would hinder skimming, the attacker would still be able to passively eavesdrop when the shielding is removed during legitimate reader-to-token communication.

Assuming there are easier ways to obtain the victim's photo than brute forcing the BAC key, it could be argued that BAC is sufficient to deter attacks on the current passport, as it observes the principle that computational effort is much greater than the value of the data. This situation could change if additional private data is added to the document. The Schengen countries intend to use a scheme with additional security mechanisms, called Extended Access Control, to protect biometric data, such as fingerprints, in passports [15,97]. It is however a possibility that other personal documents, like employee ID cards, might contain weak access control and encryption even when containing information such as biometrics.

3.3 Eavesdropping

An eavesdropping attack occurs when an attacker can recover the data sent during a transaction between a legitimate reader and a token, which requires the attack to be set up in the vicinity of a likely target. The attacker needs to capture the transmitted signals using suitable RF equipment before recovering and storing the data of interest. The degree

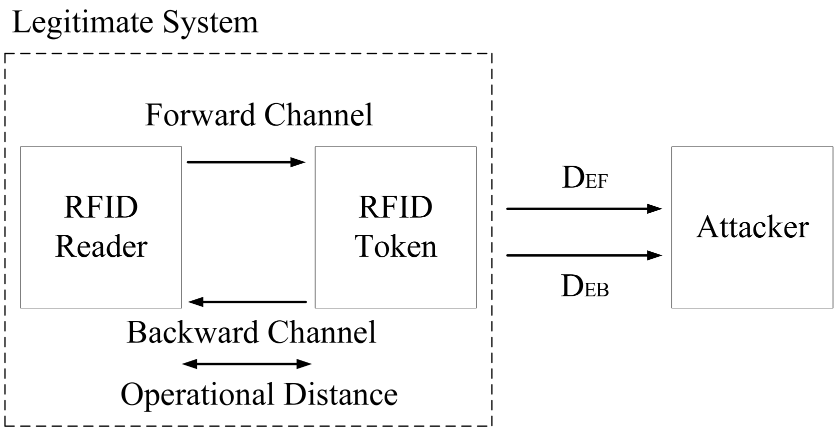


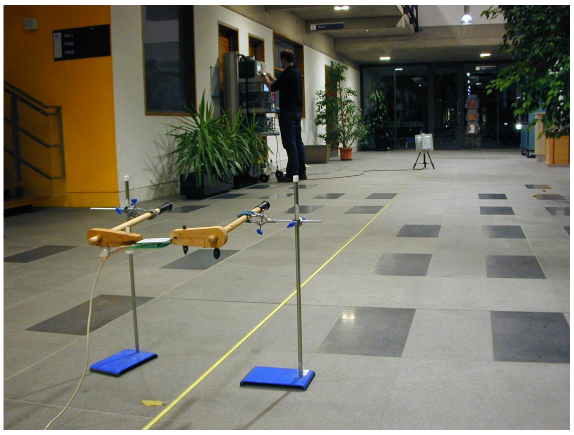
Figure 3.2: Different distance parameters of a passive eavesdropping attack

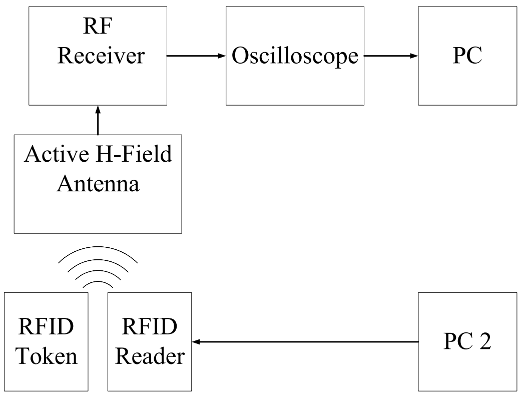
of success that the attacker will achieve depends on the resources available to him. An attacker with expensive, specialized RF measurement equipment will be able to eavesdrop from further away than an attacker with a cheap, home-made system. The attack is still a viable threat either way. An opportunistic attacker could possibly recover the credit card details of the person standing in front of him at the cashier if he had a small, portable system that could eavesdrop at 50 cm. Alternatively, if the attacker is able to successfully eavesdrop the communication from 10 m he could sit in a vehicle outside his local corner store and record all the transactions conducted inside.

As I mentioned earlier there are different eavesdropping distances to consider. Near-field communication generally uses different modulation schemes for the forward channel (reader→token) and the backward channel (token→reader). This means that the eavesdropping ranges for each of these channels are different. I therefore define D_{EF} as the distance at which the forward channel can be observed and D_{EB} as the distance at which the backward channel can be observed. The data transmitted depends on the specific application but the attacker is typically more interested in the backward channel. The exceptions are when an attacker simply wishes to determine whether a transaction took place, in which case he only needs to recover the channel with the greatest eavesdropping distance, or when information on the weaker backward channel is echoed on the stronger forward channel. For the purpose of my work I assume an eavesdropping attack to be successful at a certain distance when both the forward and backward channels can be observed at this distance.

3.3.1 Experimental setup

I set up a simple eavesdropping attack as shown in Figure 3.3. The reader and the token were placed in clamps and the antenna positioned at the same height on a tripod so that all three loops were in the same horizontal plane. The antenna, which was connected to the RF receiver, was kept stationary while the reader and token were moved further away. Data signals from the receiver were captured using an oscilloscope and read into Matlab where further DSP functions were performed to recover the data. It should be noted that a number of factors, as discussed later in this section, affect the results of an eavesdropping attack. As a result this work is not about establishing a maximum eavesdropping distance


(a) Observing and capturing communication



(b) Experimental setup

Figure 3.3: Setup for the eavesdropping experiment

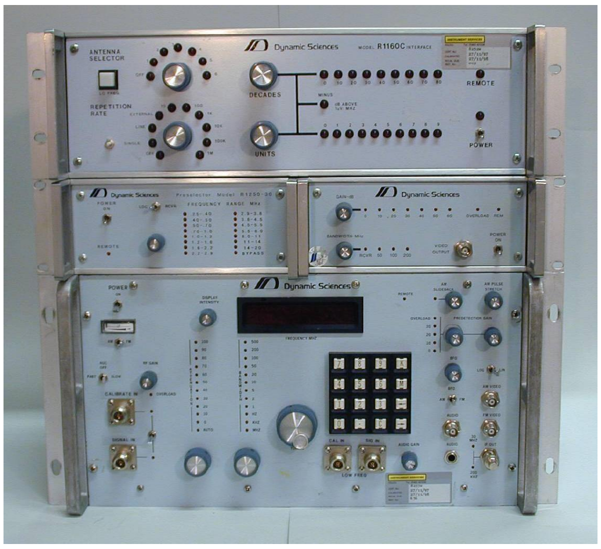
but rather about practically implementing a proof-of-concept attack using a documented method that can be re-created by other researchers to obtain comparable results for their specific environment.

Equipment

There are commercial RF receivers available that can be used to demonstrate the eavesdropping attack. I used the R-1250 Wide Range Receiver and the R-1150-10A Portable Antenna Kit, both manufactured by Dynamic Sciences. The R-1250 is a superheterodyne receiver operating from 100 Hz to 1 GHz with 21 selectable bandwidths, increasing in steps of 1-2-5 from 50 Hz to 200 MHz, centered around 200 kHz or 30 MHz IF frequencies. The receiver allows the user to adjust the RF and pre-detection gain over 50 dB and 30 dB respectively. The user can then choose whether to use the AM, FM or IF output available. Detailed information about the R-1250 receiver, including calibration data for the specific receiver used in the attack, can be found in [99, pp 23–33]. The antenna kit includes a set of H-field ferrite core antennas for field-strength measurements in the 100 Hz to 30 MHz range. Looking at the H-field is of particular interest when taking into account the dominance of the H-field in the near-field of loop antennas. The receiver is shown in Figure 3.4(a) and the antenna can be seen in Figure 3.4(b).

Currently there are three popular standards for passive near-field devices operating at the frequency of 13.56 MHz: ISO 14443A, ISO 14443B and ISO 15693. I described each of these standards in Chapter 2. Since each standard has a different communication scheme it would not suffice to make claims about eavesdropping HF devices without investigating all the standards.

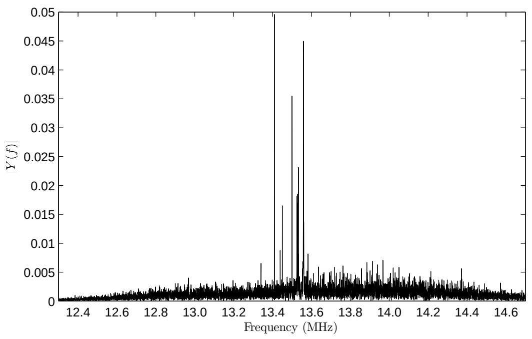
For the eavesdropping experiment I used the ACG Multi-ISO RFID Reader (Antenna dimension: 9 cm × 6 cm). I then used the following tokens: NXP Mifare Classic [122] for ISO 14443A, contactless credit card for ISO 14443B and NXP I-Code [73] for ISO 15693. I would like to point out that I used these products because they were good examples of different HF systems implemented today using the three main HF RFID standards. I do not wish to imply that any of these products are more at risk of eavesdropping than another comparable product.


(a) RF receiver [99]

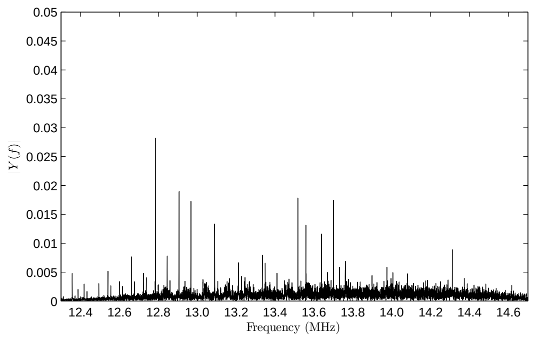
(b) Active H-field antenna

Figure 3.4: Commercial RF testing equipment

Environment



(a) Main entrance hall



(b) Hardware lab corridor

Figure 3.5: Comparative frequency-domain representations of background noise in two locations (RF Receiver: f_c=13.56 MHz, BW = 2 MHz)

It is expected that the magnitude of the H-field will decrease rapidly in the near-field, d$\leq \lambda_{f_c} \cdot \frac{1}{2\pi} \approx 3.5$ m, proportionally to $\frac{1}{d^3}$. At larger distances the decrease in the H-field will be proportional to $\frac{1}{d^2}$. The eavesdropper requires a favourable signal-to-noise ratio (SNR) to recover the data. The nature of the background noise will therefore affect the eavesdropping distance. This experiment was not performed in an empty, shielded chamber but in a laboratory that houses equipment that might emit RF signals, or contain metal, which could interfere with the magnetic field originating from the reader. Figure 3.5 shows the frequency characteristics of the background noise for two possible eavesdropping locations: The main entrance hallway of the Computer Laboratory and the corridor outside our group's hardware laboratory. The average power of the input signal to the receiver in both cases is approximately -86.5 dBm. One obvious difference between the environments that I would like to comment on is the spectral peaks around 13.5 MHz that can be observed in Figure 3.5(a), which is most likely the result of several ISO 14443 door access control readers located at regular intervals throughout the entrance hallway.

Apart from the background noise there are several other practical factors influencing the eavesdropping environment. The antenna size and transmitted power depend on the specific reader used in a system. At the same time the coupling between the token and reader also influences the eavesdropping distance as it affects the carrier amplitude and the modulation index of the backward channel. These variations are not easy to quantify since any loop antenna or oscilloscope probe used to measure these values will also influence the system. Similarly, the orientation and the proximity of the card to the reader can also affect the eavesdropping range [58].

3.3.2 Method

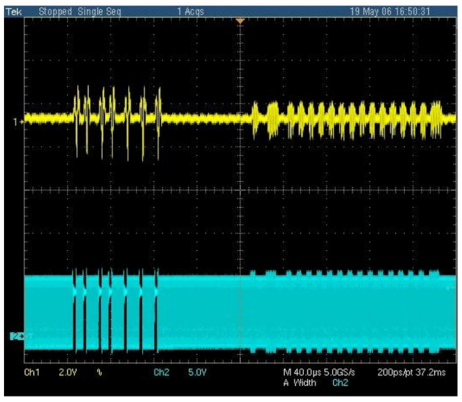
The main goal of my experiment was to show that eavesdropping on HF RFID devices are possible at non-trivial distances. As mentioned already there are multiple environmental variables to consider. Since it was not feasible to try all possible variations I limited my experiment to a single reader and three tokens adhering to different operating standards. Secondary goals were to determine to what extent the different modulation schemes influenced the eavesdropping range and to investigate whether data could be reliably recovered from a recording with a low SNR. The experiment was repeated in two different locations as discussed in the previous section.

Reference data

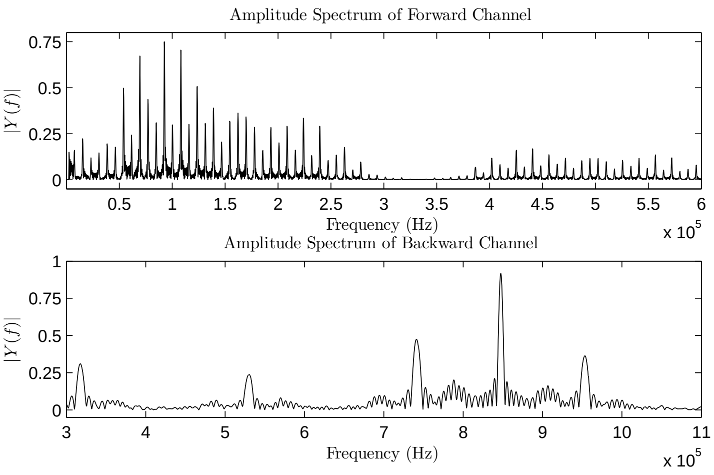
The first step of the eavesdropping experiment was to generate a set of reference data for later comparison to the recovered data, and to identify the frequency bands of interest. To generate reference data I required a transaction where the data transmitted on the forward and backward channel was repeatable. The standards in question all have a command instructing the token to return a unique identifier, which was ideal as the data always stayed the same. I recorded the signal at the antenna of the reader and demodulated it to obtain the reference data. I also computed the frequency spectrum for the forward and backward channels using the Fast Fourier Transform (FFT) of this data. If the data were still modulated onto the HF carrier the origin of the calculated spectrum will shift to 13.56 MHz.

ISO 14443A: The reader transmits 106 kbit/s Modified Miller encoded data using 3 μs pulses. The forward channel data should therefore be in the first 330 kHz of the spectrum. The token transmits 106 kbit/s Manchester encoded data, which is ASK modulated onto a 847 kHz subcarrier. The backward channel should be in a 424 kHz band centered around 847 kHz. The forward channel is amplitude modulated onto the 13.56 MHz carrier with a modulation index of 100%, while the backward channel has a modulation index of 8–12%. Figure 3.6 shows the modulated carrier, the AM demodulated output of the RF receiver and the relevant single-sided frequency spectra for a communication sequence example.

ISO 14443B: The reader transmits 106 kbit/s NRZ encoded data. The forward channel data should therefore be in the first 106 kHz of the spectrum. The token transmits 106 kbit/s NRZ encoded data, which is BPSK modulated onto a 847 kHz subcarrier. The backward channel should be in a 212 kHz band centered around 847 kHz. The forward

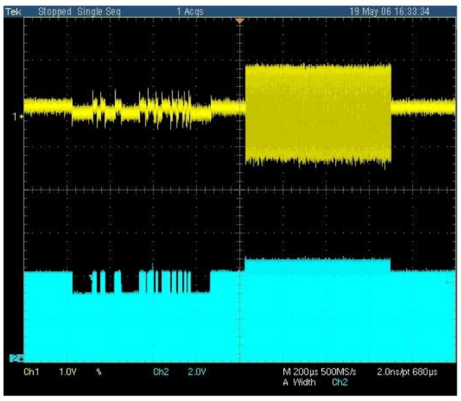

Tek Stopped Single Seq 1 Acqs 19 May 06 16:50:31
Ch1 2.0V Ch2 5.0V M 40.0µs 5.0GS/s 200ps/pt 37.2ms
A Width Ch2

(a) Time domain: Forward (left) and backward (right) channels

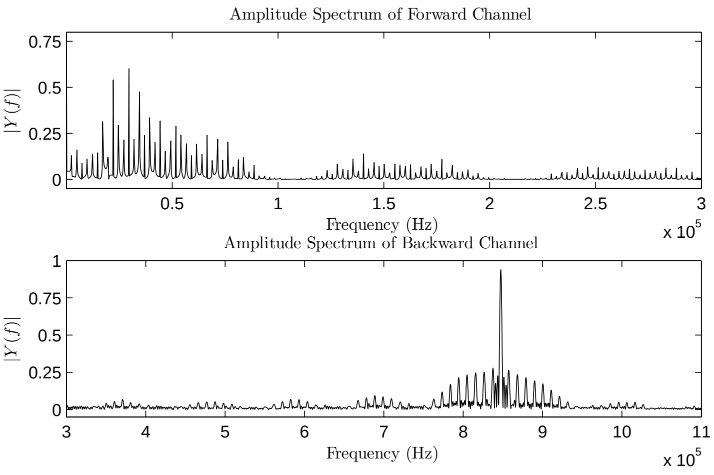


(b) Frequency domain after the 13.56 MHz carrier has been removed

Figure 3.6: ISO 14443A communication


Tek Stopped Single Seq 1 Acqs 19 May 06 16:33:34
Ch1 1.0V Ch2 2.0V M 200µs 500MS/s 2.0ns/pt 680µs
A Width Ch2

(a) Time domain: Forward (left) and backward (right) channels

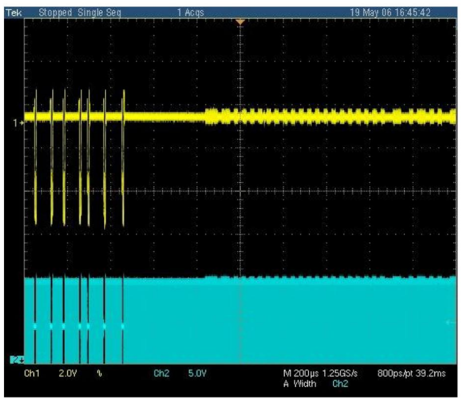


(b) Frequency domain after the 13.56 MHz carrier has been removed

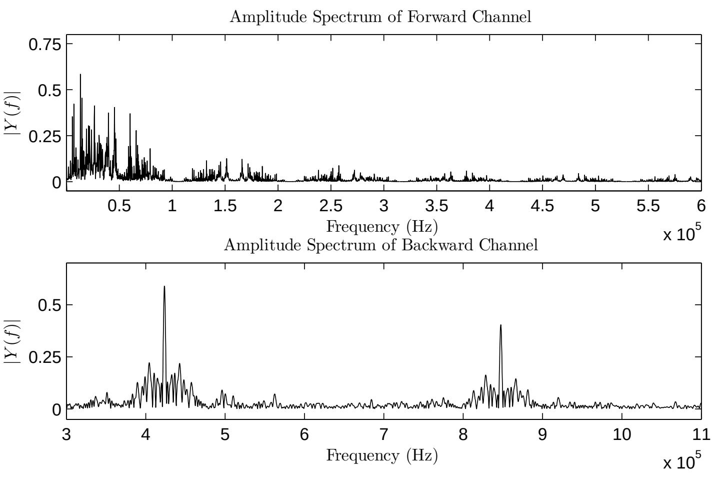
Figure 3.7: ISO 14443B communication

channel is amplitude modulated onto the 13.56 MHz carrier with a modulation index of 10%, while the backward channel has a modulation index of 8–12%. Figure 3.7 shows the modulated carrier, the AM demodulated output of the RF receiver and the relevant single-sided frequency spectra for a communication sequence example.

ISO 15693: The reader uses a '1 of 4' PPM code with a 9.44 µs pulse to transmit 26.48 kbit/s data. The forward channel data should therefore be in the first 106 kHz of the spectrum. The token transmits 26.48 kbit/s NRZ encoded data, which is ASK modulated onto a 423 kHz subcarrier. The backward channel should be in the 53 kHz band centered around 423 kHz. The forward channel is amplitude modulated onto the 13.56 MHz carrier with a modulation index of 10%, while the backward channel has a modulation index of 8–12%. Figure 3.8 shows the modulated carrier, the AM demodulated output of the RF receiver and the relevant single-sided frequency spectra for a communication sequence example.


Tek Stopped Single Seq 1 Acqs 19 May 06 16:45:42
Ch1 2.0V Ch2 5.0V M 200μs 1.25GS/s 800ps/pt 39.2ms
A Width Ch2

(a) Time domain: Forward (left) and backward (right) channels



(b) Frequency domain after the 13.56 MHz carrier has been removed

Figure 3.8: ISO 15693 communication

Capturing and calibration

The second step was to capture the signals with the RF receiver and record them on the oscilloscope. During early experiments [61] I triggered the oscilloscope on the serial communication between the host PC and the reader. I later decided to change this method as it was not an accurate reflection of an attacker's actions. There was also a possibility that the additional cables connected to the reader could aid signals of interest to radiate, thereby providing an inaccurate result. Instead I captured the 30 MHz IF output of the RF receiver for a duration of 320 ms at a sampling frequency of 100 MS/s, while the reader was continuously querying the token's identifier. For each eavesdropping scenario I made two captures, the first with the receiver's center frequency and bandwidth set to 13.56 MHz and 2 MHz respectively and the second with the center frequency set to the applicable sideband, 14.4 MHz and 13.98 MHz, with bandwidths of 500 kHz and 200 kHz respectively.

The RF gain of the receiver is adjusted by turning a knob, which does not provide an accurate indication of the actual gain introduced. The relative gain of the receiver was therefore measured before each sequence capture. This was done by providing a reference signal, a center-frequency sine wave, as input to the receiver. Its power in dBm was then adjusted until the receiver's output corresponded to a chosen value on the oscilloscope: 224 mV root-mean-square for the 30 MHz IF output signal, which is approximately 0 dBm. This gain value can then be used to determine the power of the corresponding input from the antenna to the receiver.

Data recovery

The final step is to recover the data from the recorded signal. The SNR of the data decreases with distance and eventually the data can no longer be verified visually, or recovered with a simple threshold function such as a comparator with hysteresis. This does not mean that the data is lost, but that recovery requires further processing to limit the effect of the noise. A common way to reduce the effect of noise is to average several recordings of the same signal. I do not consider this option, because the attacker does not have multiple recordings as the transaction is run only once. A number of receivers optimized to recover signals corrupted by Additive White Gaussian Noise (AWGN) have been proposed, such as the correlation or matched-filter receivers [129, pp 233–244]. The correlation receiver uses N correlators, which projects the received signal $r(t)$ onto N base functions $f_k(t)$.

$$y_k = \int_0^T r(t) f_k(t) d_t, \;\; k = 1, 2, \ldots, N$$

The matched filter receiver uses N linear filters with impulse response $h_k(t) = f_k(T - t)$, to achieve a similar output.

$$\begin{gathered} y_k = \int_0^t r(\tau) h_k(t-\tau) d_\tau \\ y_k = \int_0^t r(\tau) f_k(T - t + \tau) d_\tau \\ y_k = \int_0^T r(\tau) f_k(\tau) d_\tau, \;\; k = 1, 2, \ldots, N \end{gathered}$$

It should be noted that if the base function is rectangular then $f_k(t) = f_k(T - t)$ for the matched filter. In this case the correlator also becomes an integrator.

$$y_k = \frac{1}{\sqrt{T}} \int_0^T r(t) d_t,$$

I used a correlation receiver to recover data from the stored noisy signal. For each of the standards' forward and backward channels $N = 1$ and the base function is rectangular. The only important parameter is T, which was assigned the following values:

- ISO 14443A: Forward channel $T = 3$ µs, backward channel $T = \frac{1}{212 \text{ kHz}} = 4.72$ µs.
- ISO 14443B: Forward channel $T = \frac{1}{106 \text{ kHz}} = 9.44$ µs, backward channel $T = \frac{1}{106 \text{ kHz}} = 9.44$ µs.
- ISO 15693: Forward channel $T = 9.44$ µs, backward channel $T = \frac{1}{52.96 \text{ kHz}} = 18.88$ µs.

An example for recovering the data on the backward channel for ISO 14443A is shown in Figure 3.9. The process is as follows: (a) is the noisy signal, (b) is the data after it has been filtered using Finite Impulse Response (FIR) filters. The next step is to demodulate the sub-carrier. For ASK I rectified the signal shown in (c) before correlating it with the base function. (d) is the correlator output, which is then sampled to obtain the Manchester encoded data (e). The Manchester data is decoded to NRZ and compared to the reference data. The ISO standards define a strict bit-period grid, relative to the last bit sent by the reader, in which the token's response must be sent. The sampling times can therefore be derived from the forward channel data. Alternatively, a clock recovery scheme as described in [99, pp 125] can be implemented. The attacker can use known data, e.g. *ATQA* and *SAK* responses, to optimize his sampling thresholds, etc.

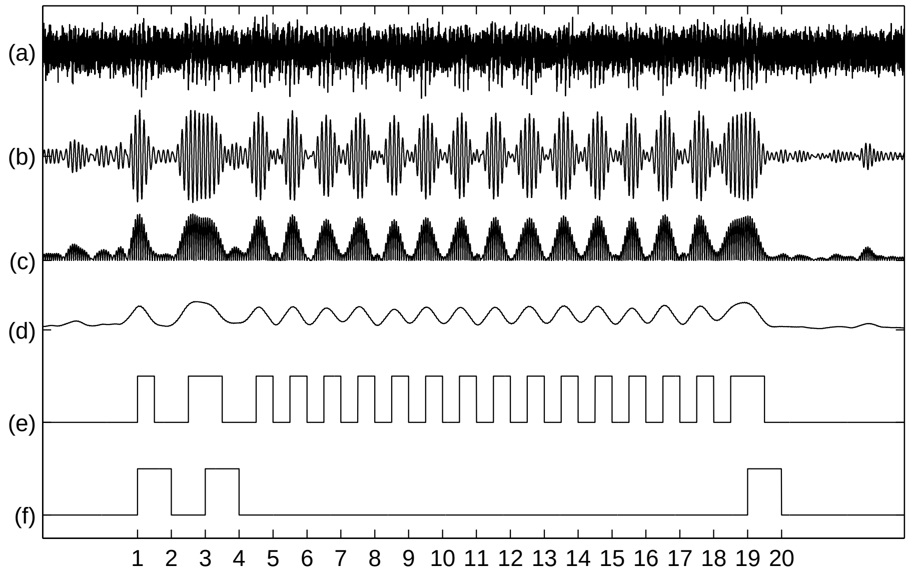


Figure 3.9: Recovering the data from a noisy signal

3.3.3 Results

Before presenting my results I first discuss the details of the eavesdropping test described in [58]. This test uses a NXP Pegoda ISO 14443A reader and seven different ISO 14443A tokens from 4 manufacturers. The authors use a matched loop antenna and a 'receiver system' (unspecified whether commercial equipment or custom build) in addition to an oscilloscope and a protocol analyser to capture a token's ID. A high level functional diagram of the receiver is provided but no details are given about the filters, amplifiers and IF sections shown. An eavesdropping attempt is considered successful when the receiver's output has a SNR greater than 6 dB, which is the level needed by the protocol analyser to obtain the correct ID. The experiment is performed with two different antenna setups: All three loops centered around the same horizontal axis, which resulted in eavesdropping distances of 5–6.5 m, and all three loops in the same horizontal plane, the same as my setup, which resulted in eavesdropping distances of 8–15 m. The fact that seven tokens, adhering to the same standard and communicating with the same reader, yield different results is a good example of how eavesdropping distances vary depending on the specific system components.

My results are shown in Table 3.1. Even with additional signal processing I did not manage to achieve the distances in [58], although my results for ISO 14443A tokens are similar to those presented in [53]. There are, however, some interesting conclusions. The forward channel of the ISO 14443A and ISO 15693 communication can be eavesdropped at a much greater distance than the backward channel, but for ISO 14443B D_{EB} is greater than D_{EF}. In addition, it is once again shown that results can vary for different locations since the ISO 14443B forward channel and ISO 14443A backward channel could be recovered in one location, but not the other.

There is still scope for further work on RFID eavesdropping, such as testing different readers and developing better data recovery methods. I started doing some preliminary work on testing how the tuning of the reader and the token affects the eavesdropping range. I placed the antenna 1 m away from the reader and displayed the AM demodulated output

	ISO 14443A	ISO 14443B	ISO 15693
Entrance hall			
1 m	FB	FB	FB
2 m	FB	FB	FB
3 m	Fx	xB	Fx
4 m	Fx	xx	Fx
5 m	Fx	xx	Fx
Lab corridor			
1 m	FB	FB	FB
2 m	FB	FB	FB
3 m	FB	FB	Fx
4 m	Fx	xB	Fx
5 m	Fx	xx	Fx

Table 3.1: Eavesdropping results: F – Forward channel recovered, B – Backward channel recovered.

of the RF receiver on the oscilloscope. By changing the parallel tuning capacitor value on the reader the amplitude of the backward channel data recovered by the receiver could be largely reduced. This also decreases the operational distance, although this might be an acceptable sacrifice to limit the risk of eavesdropping.

Finally, it is interesting to note that the ISO 18092 "Near-Field Communication (NFC)" standard, described in Chapter 2, prescribes the same modulation scheme as ISO 14443A. Devices can operate in *passive* mode, where one device acts as a reader and the other as a token, as well as in *active* mode, where both devices act like a reader. In *active* mode the devices take turns to transmit data using 100% ASK modulation of their respective carriers, effectively creating a 'forward' channel in both directions. Such a system could possibly be more vulnerable to eavesdropping, since the eavesdropping distance would be equal to D_{EF}.

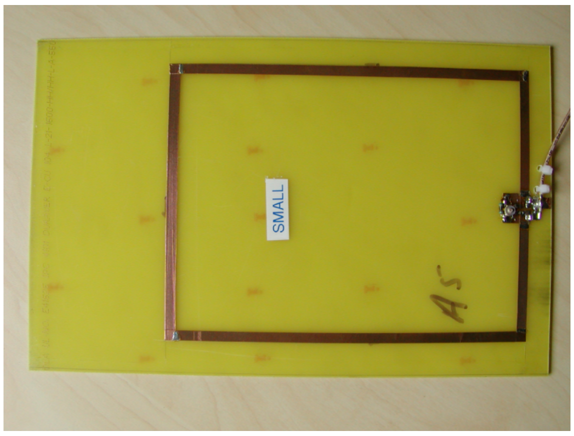
3.3.4 Eavesdropping attacks in the real world

An attacker can execute an eavesdropping attack if he acquired a suitable antenna, an RF receiver and a method to sample and record the data. Even though I illustrated the eavesdropping attack using commercial RF equipment I also want to point out that these attacks can work outside 'laboratory conditions' with cheap and portable hardware.

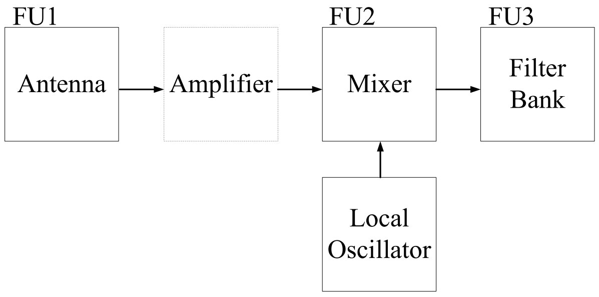
Receiver

The RF receiver converts the modulated HF carrier to a chosen IF after which the signal is filtered to isolate the frequency components that are of interest. The use of RF mixers is well documented, e.g. [128], and detailed reference designs for receivers are publicly available, e.g. [126]. A diagram showing the main Functional Units (FU) of such a generic RF receiver is provided in Figure 3.10.

FU1 – Antennas: A number of sources describe how to build HF antennas for receiving RF signals, e.g. [27, 94]. Unfortunately these concentrate mainly on E-field antennas for radio applications, although some practical construction and tuning tips still prove useful.



(a) Passive loop antenna



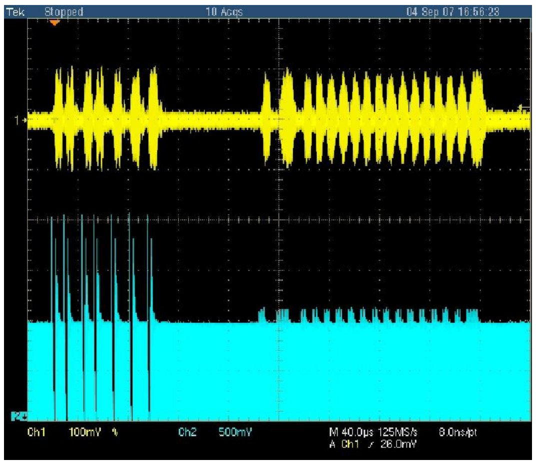
(b) Receiver functional diagram

Figure 3.10: Components of an eavesdropping receiver

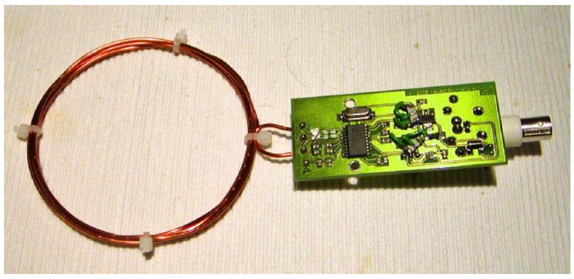
The simplest option for building a magnetic antenna is to implement one of the reference designs from TI's Antenna Cookbook [160], since most of the matching components and construction material are already specified. Alternatively, any form of loop antenna can be implemented and then matched using the guidelines in [161]. It should be noted that these guidelines specify components with a higher power rating, since the antennas are also intended for transmitting. When the antennas are only used to receive signals, components with less stringent power requirements can be used. Enameled copper wire and adhesive copper tape can easily be used to construct HF loop antennas of different sizes and number of loops. An antenna made with adhesive copper tape wire is shown in Figure 3.10(a). The resonant antenna also acts as a crude bandpass filter around the chosen center frequency. The width of the passband can be adjusted by changing the Q-factor.

FU2 – Mixer: An optional amplifier stage can be added between the antenna and the mixer. The amplifier's gain depends on the intended range of the receiver, i.e. short range protocol analyzer or longer range eavesdropping, although it should be kept in mind that most commercial mixer ICs expect an input signal with smaller amplitude and some ICs also have integrated amplifiers. The mixer's function is to move a spectral band of interest to a chosen intermediate frequency (IF) through direct downconversion. Normally, the advantage of IF systems is that any input signal can be moved to a single IF frequency by using an adjustable mixing frequency, which simplifies the design of the filter bank. In our case the local oscillator's frequency can be fixed, but using an IF still simplifies the filter implementation since this allows the use of off-the-shelf filters designed for other applications. It is also possible to implement zero-IF receivers that mixes the input down to the baseband (0 Hz). A lowpass filter can then be used to remove the unwanted high frequency components.

FU3 – Filter bank: Filtering helps to isolate the data of interest and remove unwanted frequency components. The filter bank implementation depends on the IF chosen. Choosing an IF that is often used in radio systems, like 10.7 MHz, simplifies the implementation since suitable filters can be purchased. If the system needs to work at another IF it will require the design of custom filters. Information on filter design and relevant tools can be found from most of the large semiconductor manufacturers, e.g. [3, 116, 159]. It should be noted that both passive and active high-frequency filters are sensitive to stray capacitance, or inductance, introduced by the circuit layout. The operational amplifiers selected for use in the active filters must also have adequate slew rate and gain bandwidth to function at the chosen IF.



(a) Trace of ISO 14443A *REQA* command



(b) Inexpensive RF receiver

Figure 3.11: Details of a homemade eavesdropping kit

It is possible to design and construct an RF receiver that could be used to observe both the forward and backward communication of an HF RFID system for less than £50. Figure 3.11(a) shows an example of ISO 14443A data recovered with an RF receiver, based on an existing design [126], shown in Figure 3.11(b). The receiver mixes the 14.40 MHz upper sideband down to an IF of 10.7 MHz before using a 500 kHz band-pass filter to recover the sideband data and attenuate the strong carrier. The filter also passes some higher harmonics of the forward channel data. The forward channel pulse shapes are distorted although they are still in the correct position, which is enough information to recover the data in this case. This receiver did not achieve the same results as the commercial RF receiver but I managed to recover the communication on both the forward and backward channels at a range of 60 cm, with no additional amplifier between the antenna and mixer and an antenna of 10 cm radius. However, it shows that even a cash-strapped attacker can construct a suitable receiver that could be used in a real attack. In reality one should assume that an attacker may have more resources available, in other words he might be in the position to purchase commercial RF equipment.

Signal capture and demodulation

The attacker needs to capture and demodulate the signal from his receiver. The sampling rate used by the attacker is dependent on the output of his receiver, since the rate needs to be at least twice the highest frequency component of the output to prevent aliasing effects. For example, if he used a zero IF receiver with a 1 MHz low pass filter he would need to sample at 2 MHz. An attacker can choose to make a recording and perform data recovery later or implement a real-time demodulator/decoder using a fast enough FPGA or DSP device. If the attacker chose to store a recording the amount of memory needed will depend on the sampling rate chosen. For example, an attacker taking 8-bit samples at a rate of 2 MHz for 10 s would need 20 MB of memory to store each recording. This would be higher if he uses oversampling or if he needs to sample a higher IF output. These requirements are not unrealistic taken into account that an attacker can acquire suitable hardware for a few £100, since most Field Programmable Gate Arrays (FPGA) or Digital Signal Processing (DSP) development kits come with the necessary Random Access Memory (RAM) and Analog-to-Digital Converters (ADC).

3.4 Skimming

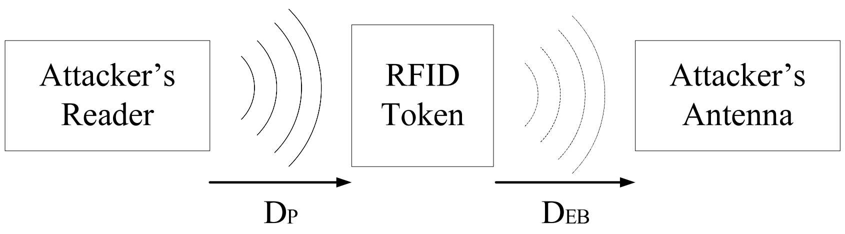


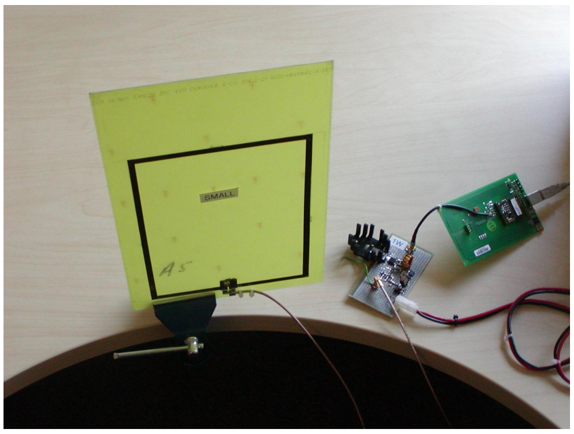
Figure 3.12: Different distance parameters for a skimming attack

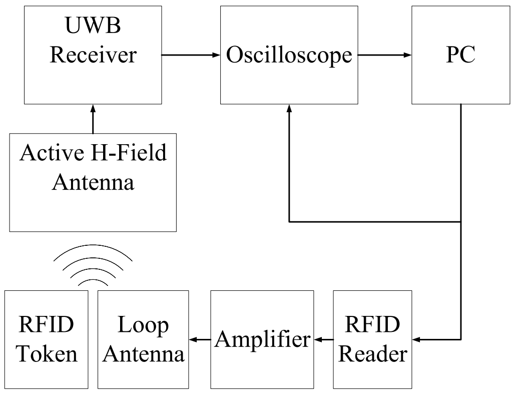
The skimming attack occurs when an unauthorized reader gains access to data stored on a token. In this attack scenario an attacker tries to read the token without the victim knowing. As is the case with eavesdropping, there are different skimming distances to consider. The attacker needs to provide power and send commands to the token, which he can achieve at a distance of D_{P}. The attacker then needs to recover the token's response, which is the same as recovering the backward channel, so this distance is defined as D_{EB}. D_{P} is the furthest distance from the reader that a card still powers up and interprets a command, not the distance the token must be from the reader to allow an attacker to recover the response. If there is a single attacker the overall skimming distance is $\min(D_{\mathrm{P}}, D_{\mathrm{EB}})$.

Ideally, an attacker must increase the operational range of his reader to avoid raising suspicion. Increasing the range is not a new technical challenge and methods for doing this are in fact described in application notes by several RFID chip manufacturers, e.g. Texas Instruments [161] and ST Microelectronics [156]. The range can be extended simply by enlarging the antenna and increasing the transmitted power of the reader. An attacker, however, does have an additional advantage in executing an attack since he is not bounded by the same transmission limits [32] adhered to by industry designers.

Since most of the application notes for increasing the operational range described a reader with a single antenna, I wanted to investigate an alternative skimming setup with two antennas. In this scenario one attacker has a modified reader that activates and commands the token, while the second attacker has commercial eavesdropping equipment to recover the response. The reason I proposed this setup was that it allowed the attackers some flexibility when deciding on which antennas to use. For example, they could conceal a small loop antenna, which works well for power coupling close to the target token. A larger H-field RF antenna designed purely for receiving, such as the Dynamic Sciences H-field antenna, can then be placed further away to recover the response. I wanted to test whether this setup could achieve a greater distance D_{EB} when compared to the single antenna attack. A single antenna attack was later demonstrated by Kirschenbaum and Wool [92].

3.4.1 Experimental setup


(a) Skimming ‘reader’



(b) Experimental Setup

Figure 3.13: Setup of the skimming experiment

The RF equipment and the environment is the same as described in Section 3.3.1. The only difference is the reader, which has been fitted with an amplifier and a larger antenna, and the position of the token, which is now attached to a wooden stand at the same height as the reader’s antenna. I tested the attack with an ISO 14443A compliant Mifare Classic token. The attack setup is shown in Figure 3.13.

Skimming reader

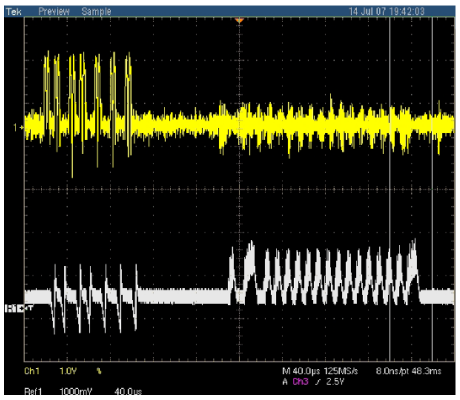
I connected the output signal of the ACG Multi-ISO reader to a power amplifier and transmitted it to the card using a copper tape loop antenna. I used four amplifiers and three copper-tape loop antennas of different sizes. The amplifiers are class E, so they are efficient for narrow-band signals and therefore well suited to RFID applications. A reference design for this class of amplifier, used to extend the range of an ST Microelectronic near-field coupling IC, is shown in [156]. I already discussed the construction of loop antennas in Section 3.3.4. The antennas were designed for Q $\approx$ 5 and capacitive matching was used to tune the antennas to the required center frequency.

Pick-up coil

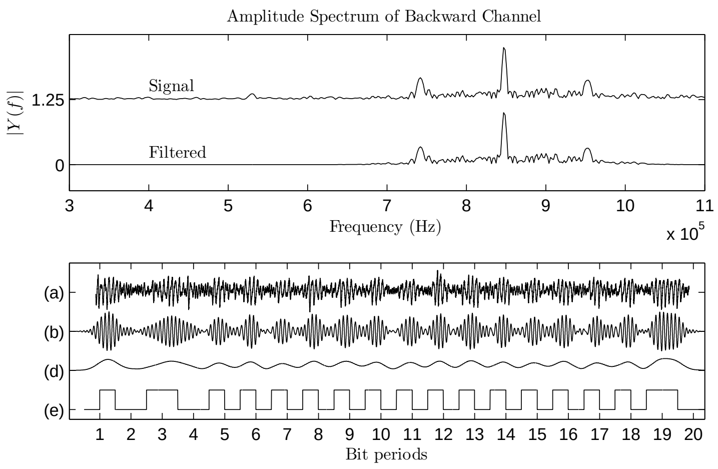
To test whether a token had been powered and received the data correctly I used a pick-up coil in close proximity to see if it generated the correct response. It is not sufficient to use the eavesdropping system as in some cases no response can be detected even though the token did respond. The pick-up coil I used consisted of a small, tuned copper loop antenna and an envelope detector, which allowed me to quickly determine the value of D_{P}.

3.4.2 Method

I first used the pick-up coil to determine the maximum activation distance D_{P} that can be achieved with different combinations of reader amplifiers and antennas. This was done by systematically moving the token further away from the reader in 1 cm increments and checking with the pick-up coil, held against the token, whether the token responds. I assumed that the token was sufficiently powered and successfully received a command from the reader if the token provided a valid response. I then left the token at this distance from the reader and tried to recover the backward channel data with the eavesdropping system. If I failed to recover the response data the token would be moved closer to the reader in steps of 1 cm until I could recover the token's response with the H-field eavesdropping antenna placed 10 cm away from the token. I chose 10 cm as a starting distance as it is the advertised operating range for ISO 14443 systems and I felt that skimming could only be deemed successful if demonstrated beyond this distance. If I succeeded in recovering the response the eavesdropping antenna would be placed further away, in multiples of 20 cm, until I failed to recover the token's response. After some initial experimentation, I estimated D_{P} to be in the 0–30 cm range and following on from my eavesdropping results I expected D_{EB} to be in the 0–400 cm range. For this reason, I chose the distance increments for the eavesdropping antenna to be larger than the distance increments for the token from the reader. Once I failed to recover the response the token was again moved closer to the reader and the process was repeated. An example of the eavesdropped signal, the pick-up coil reference and the data recovery is shown in Figure 3.14. I performed the experiment with all the different combinations of antennas and amplifiers. The reference data, signal capture method and data recovery steps are the same as those described for eavesdropping of the backward channel in Section 3.3.2.



(a) A5 antenna with 1 W amplifier, $D_{\mathrm{P}} = 15$ cm and $D_{\mathrm{EB}} = 2$ m



(b) Frequency spectrum and data recovery

Figure 3.14: Example of skimming results showing the time signal trace, the frequency spectrum and the recovered data

3.4.3 Results

The maximum distance D_{P} for each combination of antenna and amplifier is shown in Table 3.2. D_{P} increases as the antenna size and the transmitted power increases, which was the expected result.

The best result for D_{EB} was 2 m, obtained using the 14.8×21 cm antenna with the 1 W amplifier when D_{P} was 15 cm. I expected D_{EB} to increase in the same way as D_{P} did.

	0.5 W	1 W	2 W	4 W
148×210 mm($\frac{1}{32}$m^2)	15 cm	16 cm	17 cm	19 cm
210×297 mm($\frac{1}{16}$m^2)	20 cm	23 cm	23 cm	25 cm
297×420 mm($\frac{1}{8}$m^2)	22 cm	25 cm	26 cm	27 cm

Table 3.2: Maximum D_p for each antenna/amplifier setup

Instead the distance at which I could retrieve the response actually decreased. This could possibly be attributed to a number of factors:

- The amplitude of the load modulation appeared constant and not proportional to the amplitude of the carrier signal, so essentially the modulation index was decreased each time the amplitude of the carrier was increased. This is possibly due to current limiting incorporated in the token's power supply design. As explained in Chapter 2, the carrier amplitude decreases when the impedance of the token is changed, i.e. when the amount of current that it draws changes. If the carrier amplitude is increased the token needs to attenuate the carrier more to maintain the modulation index, which means that it would need to draw more current. Current limiting protects the token's circuitry but also prevents it from sinking the current required.
- As the token was moved further from the antenna the effect of its coupling decreased. The effect of the load modulation therefore decreased as D_{P} increased and even though the token could be activated at greater distances the token had to be moved closer to influence the field of the antenna.

This negated the advantage of larger antenna/amplifier combinations. With the largest antenna D_{P} and D_{EB} were approximately equal to 20 cm. This was the largest value of D_{P} at which I managed to retrieve the backward channel. When $D_{\mathrm{P}} \approx D_{\mathrm{EB}}$ the attacker gains no advantage from using two different antennas. In fact, the two antenna method offers no distance advantage over the method using only one antenna, achieving comparable results to the single antenna scenario demonstrated in [92]. It is, however, a viable alternative if the attacker needs to limit the size of his hardware. In the two antenna case he can use a smaller antenna, with a less powerful amplifier, while placing his signal capturing equipment much further away. In [92] the skimming range of 25 cm was achieved using an antenna 40 cm in diameter. Arguably an attacker can better conceal a 14.8×21 cm antenna, which allows him to hold it closer to his target while an accomplice sits up to 2 m away recovering the token's response. The threat of the skimming attack seemed slightly diminished as D_{P} ended up quite small for the best case of D_{EB}. That said, 15 or 20 cm is probably enough to execute an attack in a crowded area and easily allows reading of a token in somebody's pocket or bag.

3.5 Eavesdropping resistance with cover noise

RFID systems are vulnerable to eavesdropping, as shown in Section 3.3. Even though confidentiality can be provided by implementing suitable application layer encryption, the cost or hardware constraints might limit the amount of logic than can be accommodated. This means that some tokens contain only data storage elements with no, or limited, security mechanisms. In certain cases key exchange is not possible as the device has no cryptographic means to do so and without a shared key no data can be exchanged confidentially. The traditional way of deriving a session key from the token's unique identifier by using a master key might not be feasible since the token responds with a random identifier to prevent tracking. Key setup and management is also a challenge in systems where the token and the reader have no previous association. Even in systems with application layer security and key management, eavesdropping is still a problem, e.g. as is the case with e-passports. As explained in Section 3.2.2 there is a potential weakness in the session key algorithm, which allows an attacker to store the eavesdropped data and execute a brute force attack offline.

Any mechanism that allows two devices to exchange shared keys or makes the channel more resistant to eavesdropping, without adding to the hardware complexity of the token, would therefore be of benefit to RFID systems. It has been proposed that additive noise on the communication channel can be used to protect data, as discussed further in Section 3.5.1. Cover noise proposals for the key-less exchange of a secret are especially useful in scenarios involving devices with limited cryptographic resources and can also be applied to ubiquitous environments, where pairing and key-exchange often happen between devices that have never interacted before. A number of protocols have been suggested in the last few years that use bit-collisions, or bit-blocking, to protect an RFID token's privacy [85, 141], or as a method to exchange keys between an RFID reader and a token [29,66]. These protocols make the security assumption that tokens, or devices acting like tokens, are indistinguishable to an attacker. The authors argue that distinguishing between different devices is hard and that it would require special hardware, collusion between different attackers or 'fingerprinting' of tokens.

I show that an attacker can distinguish between a response and corresponding cover noise because of simple differences in the devices' communication. The attacker would need no more advanced equipment beyond that needed to perform an eavesdropping attack. I therefore propose an alternative implementation of current bit-blocking schemes, where the characteristics of the cover noise are chosen in such a way that it obscures differences in the devices' phase and modulation depth.

3.5.1 Related work

The idea of exchanging data securely by using characteristics of noise on the communication channel, and without the need for a shared secret, has been around for decades, following on from the work of Shannon [148]. In 1975 Wyner [171] described the 'wire-tap' model shown in Figure 3.15. The sender transmits some data $y(t)$, which is corrupted by noise $N'(t)$ and $N''(t)$ on the communication channel. The intended recipient receives $x(t) = y(t) + N'(t)$ while the attacker receives $z(t) = y(t) + N''(t)$. The basic idea is that $N'(t) \ll N''(t)$ and as a result, based on the information theory regarding noise and channel capacity, the intended recipient can recover the data while the attacker cannot. Several ideas following this model have been proposed in the RFID environment [14,33]. The problem with these proposals are that, even though these can be shown to be theoretically secure if the noise levels are sufficient, there are no practical assurances that $N''(t)$ will always be adequate to prevent an attacker from recovering the data.

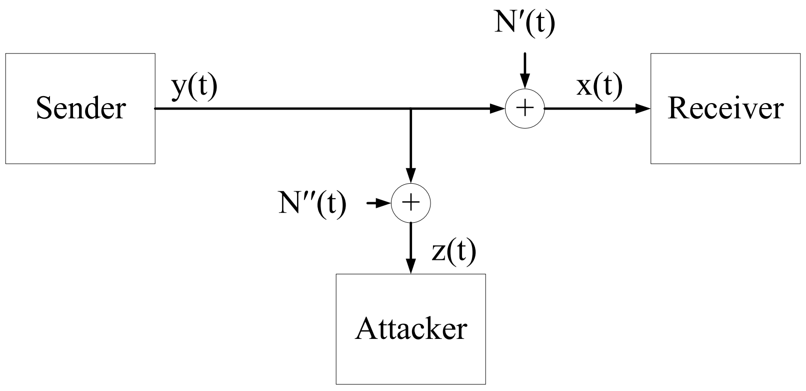
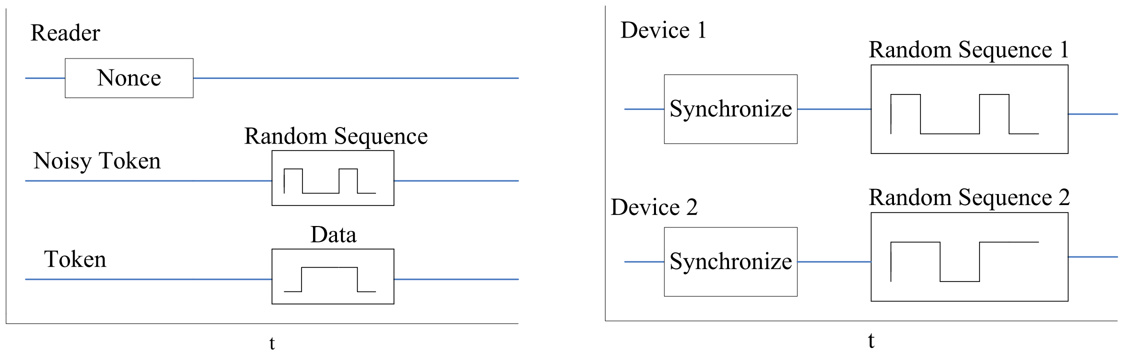


Figure 3.15: The ‘wire-tap’ channel

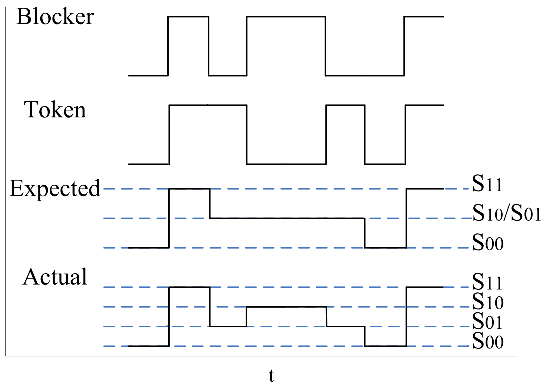
It is therefore a logical progression to intentionally add noise to the communication channel. Within the RFID environment there are several papers suggesting that a system should intentionally cause bit collisions on the channel between the reader and the token [85, 139, 141], thereby scrambling the true value of the token's response. Bit-blocking works as follows (assuming there are two devices): The devices, which are synchronized and identical in terms of their communication channel, transmit a data sequence at the same time. If both transmit a ‘1’ the result is symbol S_{11} and if both both transmit ‘0’ the result is symbol S_{00}. If the devices transmit a ‘1’ and a ‘0’ respectively the result is either S_{01} or S_{10}. Bit-blocking works on the assumption that $S_{01} = S_{10}$ and that the attacker cannot determine who sent the ‘1’ and who sent the ‘0’.



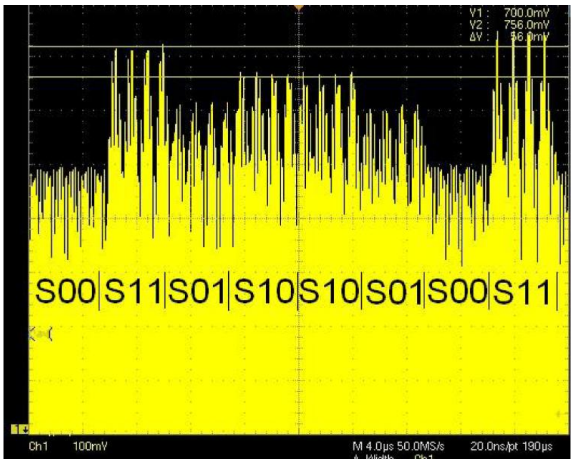
(a) Noisy Tag Protocol (NTP) (b) NFC Key Agreement (NKA)

Figure 3.16: Bit-blocking key exchange protocols

This principle can be used by token blockers to ensure privacy, or provide access control, by hiding the response from tokens in the presence of an untrusted reader [85, 141]. For the purpose of this section I concentrate more on bit-blocking as used in the key-exchange protocols, such as the Noisy Tag (NTP) and NFC Key Agreement (NKA) protocols suggested by Castelluccia, et al. [29] and Haselsteiner, et al. [66] respectively. The two protocols differ slightly in terms of practical implementation, as shown in Figure 3.16. In the NTP another token, referred to as the ‘noisy tag’, is used as the blocker. The blocker transmits a random sequence at the same time as the token responds with data intended for the reader. The reader shares a secret with the blocker, so the reader can predict the bit-blocking sequence and as a result it can recover the data transmitted by the token. An attacker does not know the bit-blocking sequence and can therefore only recover the data bits where both the data and blocking symbols are equal, i.e. when he receives symbols S_{11} and S_{00}. With NKA two NFC devices, who want to exchange a key, transmit random data sequences to each other at the same time. From the attacker's perspective Device 1 effectively acts as a blocker to Device 2, with the data sequence transmitted by Device 1 used to determine the key. During the final reconciliation stage Device 1 compares the sequence it received against the sequence it transmitted (Device 2 does the same). Both devices discard bits that would have resulted in an attacker observing S_{11} and S_{00} symbols and the secret key is refined from the data bits yielding S_{01} and S_{10} symbols. For example, if Device 1 transmits the sequence '100101' and Device 2 transmits '101000' the key is refined as '101'='10**0101**.



(a) Comparison of theoretical and practical bit-blocking



(b) Bit collision between the replies of two ISO 14443A tokens, an example showing the four different symbols

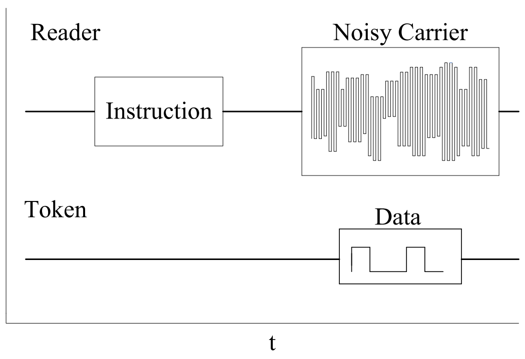
Figure 3.17: Distinguishing between cover noise and the token's response

Both the NTP and NKA protocols are useful assuming that it is practical to ensure that the $S_{01} = S_{10}$ condition holds. As the authors mention themselves, this requires that both devices' data must match in amplitude and phase. Figure 3.17(a) shows an example where S_{01} is not equivalent to S_{10}. I looked at several ISO 14443A tokens, all containing a NXP Mifare 1K IC, to see if the communication of commonly used RFID tokens varies in amplitude and phase and observed the following:

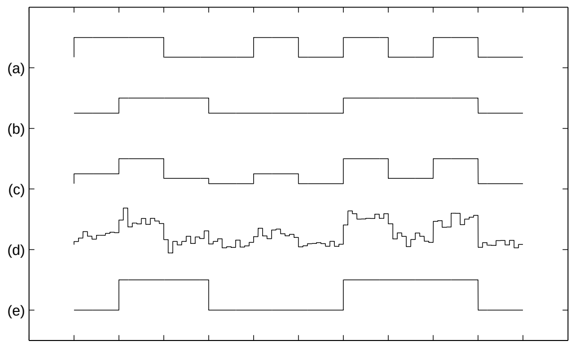
- **Amplitude**: A difference in amplitude of S_{01} and S_{10} is likely to occur if there is a difference in the modulation depth of the blocker and the token. The modulation depth, or the change in amplitude of the carrier during data modulation, is determined by the antenna inductance, the resonant capacitor, modulation impedance and even orientation (since it affects antenna coupling). Figure 3.17(b) shows that the synchronized response of two of these tokens clearly has four distinct levels for S_{01}, S_{10}, S_{11} and S_{00} respectively. As a result an attacker with eavesdropping equipment, in this case a tuned copper loop antenna and an amplifier, might be able to distinguish between the two sequences. For example, in Figure 3.17(b) the amplitude of S_{11} is about 8% greater than S_{10}.
- **Phase**: I found that phase was less of a practical issue. Tokens have the ability to synchronize relatively well, as illustrated by the anti-collision procedure in ISO

14443A cards [78]. The tokens I tested all responded within 0.1 µs of each other, which is roughly equivalent to 1% of a bit period. A determined attacker could probably fingerprint a card in this way, but at this stage variability in the amplitude of the modulation depth is an easier option.

3.5.2 Noisy carrier modulation



(a) System timing diagram



(b) Example of bit-blocking with additional noise

Figure 3.18: Noisy bit-blocking protocol

The NKA protocol suggests that each device synchronizes phase and amplitude before commencing with the rest of the protocol. In the NTP protocol a noisy token is used, which I assume is similar to the other tokens to ensure that $S_{01} \approx S_{10}$. Suggesting that two devices synchronize in time is feasible, but matching modulation depth is trickier, since it involves varying the RF carrier or tuning parameters. An alternative solution is to introduce randomness to the communication. This can prevent the attacker from determining whether the symbols S_{01} or S_{10} were transmitted. Randomizing the characteristics of the physical communication medium, by continuously moving devices involved in key pairing, have been suggested to prevent an attacker from distinguishing between devices using received signal strength [30]. In the RFID environment, however, it is hard to move the reader, so I had to look for another solution.

I propose that the blocker uses a layer of band-limited AWGN in addition to bit-blocking to hide differences in the physical characteristics of the tokens' communication. Figure 3.18(b) shows an example of how this works: (a) and (b) are the blocking sequence and data and (c) is the combination of the two. The fact that (c) has two distinct levels for S_{01} and S_{10} is hidden by adding random noise (d), but the data can still be recovered (e). In a way this merges bit-blocking with the concept of hiding data in random noise. The exchange phase of my protocol is shown in Figure 3.18(a). This is followed by a resolution phase where S_{11} and S_{00} are discarded and a key K_T is refined from the remaining symbols. This phase is the same as described in the NTP and NKA protocols and $2n$ bits need to be exchanged to refine an n-bit secret. In practice the scheme will work as follows:

1. The reader polls for tokens in its field.
2. If a token is present it responds with a simple acknowledgment.
3. The reader asks the token to suggest a key.

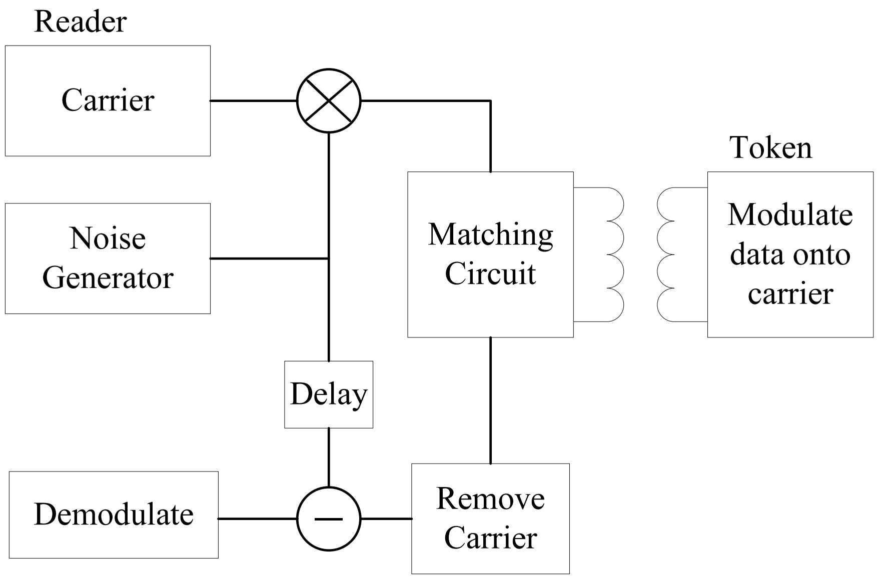


Figure 3.19: Example of practical implementation

4. The token responds with random number of length $2n$, while the reader performs noisy bit-blocking.
5. The reader and token agree on a n-bit session key based on S_{01} and S_{10}.
6. Subsequent information is encrypted using the session key.

I assume that the reader is trusted and that it attempts to exchange a key with a trusted token in the presence of a passive attacker. I do not consider active attacks and my scheme does not prevent unauthorized readers from communication with a token. It could, however, be used to exchange a secret between an RFID blocker/proxy device and a reader, which can then be used to set up authentication and access control conditions. If the user already carries an intelligent RFID proxy device, which uses bit-blocking for privacy, the scheme can be modified so that this device can randomize its blocking bits by adding AWGN, therefore making it difficult for an attacker to differentiate between the tokens and the proxy. NFC has already been advocated as an out-of-band method in ubiquitous environments for setting up communication parameters before communication commences on another medium [10]. My proposal can be extended to active devices, such as mobile phones, that use the 'passive' mode described in the NFC standard [81]. My scheme can also be used, in addition to conventional cryptography, to provide eavesdropping resistance. For example, it will make brute force key searches on e-passports much harder if some of the attacker's eavesdropped cipher text bits are incorrect. In this case the reader will transmit blocking-bits whenever the token responds with data.

Practical implementation

Practically my scheme differs from current blocker implementations. I also propose that the reader itself acts as the blocker. This makes the system simpler as the user does not need to carry an additional device, which shares a secret with all readers that are encountered. Near-field communication differs from conventional RF communication, since the token does not transmit a signal in the conventional sense. The token modulates data onto a carrier transmitted by the reader by changing its impedance [54]. Cover noise can therefore be added by generating a 'noisy carrier' onto which the token's data is modulated. The reader has a noise generator that combines the output of a PRN generator, which generates the bit-blocking sequence, and an AWGN noise source. The result is modulated onto the carrier at the same time as the token's data. After the reader removes the carrier and subtracts the noise the data can be recovered. This does not require a special token. In fact, tokens adhering to ISO standards that specify near-field communication can be used, as the bit-collision process is transparent to the token. The reader only requires minor modifications as shown in Figure 3.19.

Implementing the cover noise in this way also provides protection against attackers who try to recover data with the help of directional antennas. When the noise is generated by a third party that is not in close proximity, e.g. a device covering the whole room, or if two devices both transmit data, e.g. the NKA protocol with 'active' NFC, then an attacker can possibly isolate the data response (or the cover noise sequence) by aiming his antenna at a specific device. In near-field communication the token's response is modulated onto the signal originating from the reader. The attacker eavesdrops this signal, not a signal from the token, so the cover noise sequence and the data response should have degraded equivalently irrespective of the spatial orientation of the reader and the token relative to the attacker. This means that the attacker has minimal chance to separate cover noise and response data because of differences in the positioning between the token and reader. In the case of 'active' NFC the very short operating distance might also make it difficult for the attacker to distinguish between the blocking and data sequence.

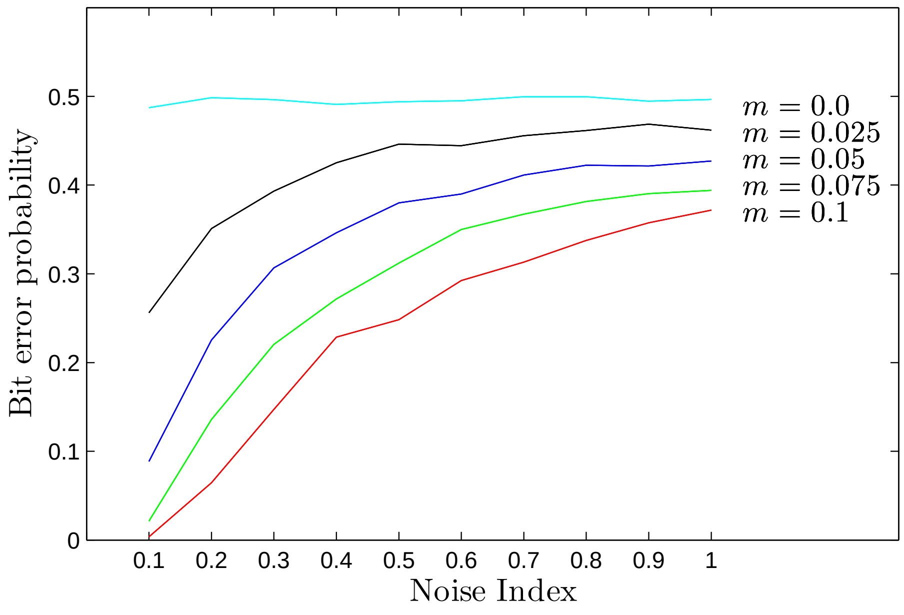


Figure 3.20: Simulated results for noisy carrier modulation

3.5.3 Results

The attacker does not know the noisy-bit blocking sequence, so he has to try and recover the data by removing the noise through alternative means. For the simulation I integrate over an entire bit period and make a decision about the symbol based on the result, as I did when recovering the eavesdropped data in Sections 3.3.3 and 3.4.3. This is an optimum receiver used for data recovery in the presence of AWGN, so it works well for testing the effectiveness of the noisy addition. I assume that the attacker knows exactly when the data is sent and that he can guess the bit period for each symbol without performing clock recovery. The attacker discards symbols S_{11} and S_{00} and calculates K_{A} based on his knowledge of S_{10} and S_{01}.

For modeling, the maximum value of the larger symbol is set equivalent to 1 and the value of the smaller symbol is set to $1-m$, e.g. in Figure 3.20 $S_{10} \approx 700$ mV and $S_{01} \approx 660$ mV, so $m \approx 40$ mV ≈ 0.055. The sequence of S_{10} and S_{01} symbols is defined as $S(t)$. The random noise $N(t)$ is generated in the range $[-1; 1]$ and scaled by a noise index n. The sequence recovered by the attacker is therefore $S_N(t) = S(t) + N(t) \cdot n$.

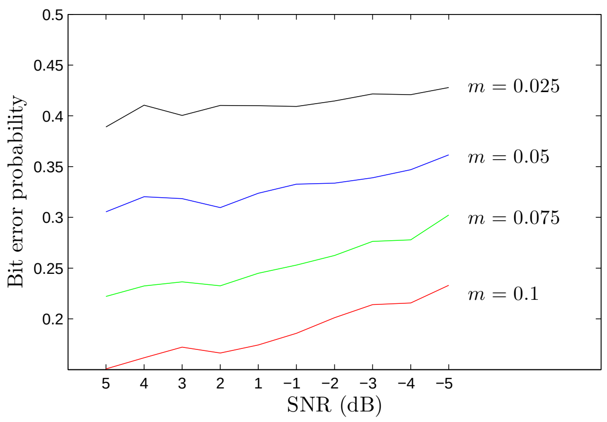
Figure 3.20 shows some results for my scheme: I calculate the probability of the attacker making a bit error and plot this against the noise index n for varying amplitude differences m. The noise frequency is ten times the data frequency and I also assume the best case for the attacker in terms of environmental noise, so there is no additional $N'(t)$. A bit error rate of 50% is equivalent to the attacker randomly guessing the key bits, as statistically he should get half of his guesses correct. The final bit-error probability for each (n, m) pair is the average bit-error probability of 100 trials, each containing 100 S_{01} and S_{10} symbols. It is not valid to assume that the data will effectively be blocked whenever $N(t) \geq S(t)$. For example, when $N(t) = S(t)$ the SNR is only 0 dB, which is still considered to be a good quality signal in signal processing terms and will allow for the recovery of data.

Apart from the amplitude of the additive noise and the amplitude difference between S_{01} and S_{10}, the frequency of the additive noise and the environmental noise $N'(t)$ can also influence the scheme's success. The best result is achieved when the noise frequency is equal to the data frequency, in other words if both are the same bit shape albeit with different amplitudes. Increasing the frequency of the generated noise, if anything, decreases the attacker's bit error probability, as shown in Figure 3.21(b), which is expected since higher frequency noise can be more effectively filtered out. Using noise frequency that is higher than the data frequency does however aid in masking the phase differences between the blocking and data sequences. As is also expected, any environmental noise that is added to the signal observed increases the probability that the attacker will make a bit error. An example of the environmental noise's effect is shown in Figure 3.21(a).

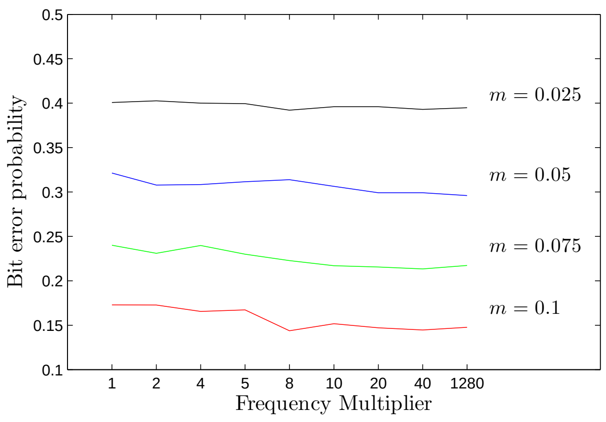
3.6 Conclusion

HF RFID devices using near-field communication are used in a number of secure application such as e-passports and credit cards. The RF communication interface of these devices are vulnerable to eavesdropping and skimming attacks. These attacks are a well known risk for RFID devices, yet few publications give details about possible experimental setup or practical results.

In this chapter I present results from practical proof-of-concept eavesdropping and skimming attacks implemented against HF RFID devices. I successfully performed eavesdropping attacks against devices implementing the three most popular HF standards: ISO 14443A/B and ISO 15693. I also demonstrate an alternative skimming attack method where the attackers use two separate antennas to power the token and retrieve its response. In each case I describe the equipment needed and document the attack setup



(a) Effect of additional environmental noise $N'(t)$ with noise index equal to 0.3



(b) Effect of the generated noise's frequency with noise index equal to 0.3. The noise multiplier is applied to the frequency of the data

Figure 3.21: Effect of noise frequency and environmental noise on the noisy carrier scheme

and execution. I also describe the implementation of an RFID receiver kit that could be constructed for less than £50, which can be used to observe RFID communication. Even though the self-build RF receiver did not achieve the same results as commercial equipment it does illustrate that eavesdropping is not beyond the means of the average attacker.

These attacks are dependent on a variety of factors so someone else with different RF equipment and environmental conditions might achieve a different result. In the attacker's perfect world, or with 'advanced monitoring equipment and ideal environmental conditions, including optical line of sight transmission, low humidity, and no radio interference', to quote [118], eavesdropping could be possible at much greater distances as is indeed shown to be the case in [58]. My main contribution was therefore not so much the actual attack distances, but rather the experimental setup that provides other researchers with a reference attack, which they can improve upon. That said, my results do show that near-field devices are not rigidly location limited and that an attacker can definitely recover data beyond the advertised operating range. It also provides a practical result to debate, which is important for RFID technology where attack distances are so often seen as a measure of security.

Finally, I propose a method for making near-field communication resistant to eavesdropping where the reader transmits a noisy carrier to obfuscate the backward channel. I show that current bit-blocking schemes used to obfuscate RFID data are vulnerable because attackers can distinguish between the blocking sequence and the data based on the difference in the modulation amplitude of the blocker and the device. I improve on these proposals by randomizing the physical communication characteristics with an additional layer of AWGN. This makes it difficult for attackers to distinguish between the blocking sequence and data, even if the devices differ in terms of phase and modulation depth. I show simulated results suggesting that this method increases the probability that an attacker will make significant bit-errors when attempting to recover the data. Apart from creating an eavesdropping resistant channel the scheme could be used for key exchange between devices with limited cryptographic resources. It can also be used by RFID blocker and proxy systems to hide any differences in their communication medium compared to the tokens they guard.

In my proposal the reader itself acts as the blocker, which simplifies the system as the user does not need to carry a special blocking device. The reader transmits a noisy carrier onto which the token modulates its data. Implementing the scheme requires little additional hardware in the reader and it is transparent to the token, so it can be extended to any inductively coupled communication, e.g. ISO 14443A/B and ISO 15693. It can also be extended to any system using 'passive' NFC technology and can therefore be applied to ubiquitous computing applications where pairing and key-exchange often happen between devices that have never interacted before. The only practical constraint on the scheme is that the carrier still needs to provide the token with a stable power source. The noise index might therefore be limited depending on the card's power consumption and ability to store energy until the carrier amplitude is again large enough to charge the storage capacitor. It should, however, be noted that in practice the environmental noise and signal attenuation will be non-trivial and as a result the SNR will decrease, increasing the probability of the attacker making an error. These factors should compensate for the noise index limiting. Another scheme, specifically for ISO 14443B, also using cover noise to prevent eavesdropping on RFID systems, was published at the same time as my proposal [147].

Chapter 4

Relay attacks: Extending proximity

RFID devices are often used for proximity identification in tracking and access control systems. These systems operate on the assumption that the token is in close proximity to the reader because of the physical limitations of the communication channel. However, current RFID devices are not suitable for secure proximity identification, as an attacker can fool the system by relaying the communication between the legitimate reader and token over a greater distance than intended. In this chapter I discuss relay attacks and their relevance to RFID systems. I also describe a practical implementation of a relay attack against ISO 14443A systems and show how the attack can be extended to facilitate additional attacks.

4.1 Introduction

RFID systems use the physical constraints of the communication channel to implicitly prove the proximity of a token, since near-field communication is seen as location limited. RFID devices are therefore used in authentication systems to link a person with a location [146], or a context [7]. Similarly, these devices are used in logistic systems to track items. A typical system will have several trusted readers placed at known locations. The reader then waits until a token enters its communication range, at which time the reader identifies the token and reports the required information to the back-end system. Since the system knows the location of the reader, and assumes that the token had to be in close proximity in order to communicate, it now knows the location of the token and the item it is associated with.

In 1976 Conway [38] first described the Grand Master Chess problem where a person could play chess against two grand masters by acting as a man-in-the-middle and effectively playing them against one another. A relay attack, or 'mafia fraud' as it was first referred to by Desmedt, et al. [43], is an extension of this scenario to security protocols. With a relay attack the 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 sent to the verifier. It should be noted that the chess player can never beat both grandmasters. Similarly, the relay attack is limited when compared to conventional man-in-the-middle attacks, since the attacker cannot modify or access any data he relays unless a further flaw exists in the protocol.

In this chapter I discuss my work investigating the significance of relay attacks to systems using RFID for proximity identification. I explain how a relay attack is executed and propose several attack scenarios related to RFID systems. I also describe the implementation of a practical relay attack against ISO 14443A RFID systems. Finally, I discuss the security implications of the relay attack on current HF RFID tokens. Parts of this chapter are based on a unpublished report I authored at the start of 2005, "A Practical Relay Attack on ISO 14443 Proximity Cards" [62], and on a section in a short paper I authored at the end of 2005, "Practical Attacks on Proximity Cards" [61].

4.2 Relay attacks on RFID

RFID systems are potentially vulnerable to an attack where the attacker relays communication between the reader and a token. A successful relay attack allows an attacker to temporarily possess a 'clone' of a token, thereby allowing him to gain the associated benefits. Some RFID tokens perform mutual authentication and encrypt the subsequent communication. An attacker, however, never needs to know the plain-text data or the key material as long as he can continue relaying the respective messages. It is therefore irrelevant whether the reader authenticates the token cryptographically, or encrypts the data, since the relay attack cannot be prevented by application layer security.

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. The relay attack setup is shown in Figure 4.1. 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 will assume that it is communicating with the reader and respond. The token's response is then relayed back to the proxy-token, which will transmit the information to the reader. The reader is unable to distinguish between the real token and the proxy and will therefore assume that the token, and its associated owner, is in close proximity.

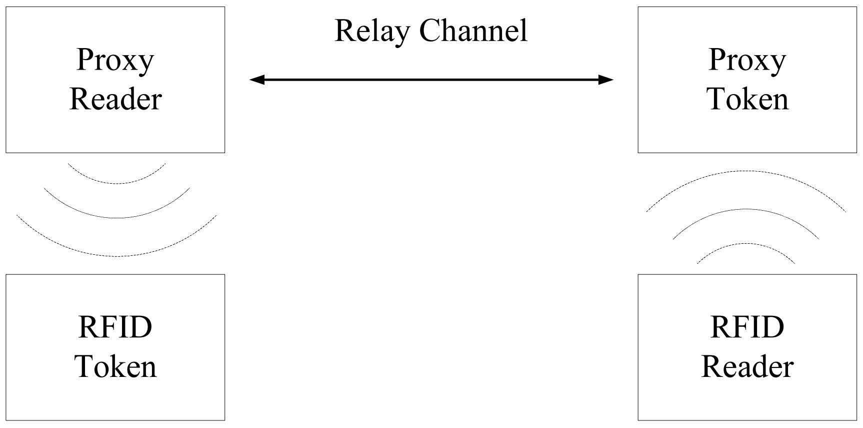


Figure 4.1: Relay attack on an RFID system

The attacker can implement his own custom hardware for the proxy token and reader, as described in Section 4.3, or alternatively use existing hardware, such as NFC devices. As mentioned in Chapter 2, the ISO 18092, or NFCIP, standard allows active devices, such as cellphones, to communicate using near-field communication. A device can act as either a reader or a token and already has 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 was presented by Kfir, et al. [90].

Relay attacks are not applicable only to RFID devices. Drimer and Murdoch demonstrated a relay attack against contact Chip-and-Pin smart cards implementing the EMV electronic payment protocol [45]. In this attack the victim had to insert his card into an attacker's card reader and enter his PIN. Once this happened, another attacker could present a proxy-card as payment to a vendor's legitimate card reader. After this attacker concluded the purchase, his accomplice returned the victim's card. There are, however, several factors that aid the attack in the RFID environment. RFID transactions seldom require further user interaction, such as a PIN. Not only does this shorten the transaction time, and therefore the time the attacker needs to have access to the token, but the attacker can possibly activate the token without the knowledge of the owner by way of a skimming attack. The attacker does, therefore, not need to convince the victim to hand over his token for a period of time. 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 the contact card attack the attacker had to present his card in the presence of the vendor, so it had to closely resemble a real card.

4.2.1 Attack scenarios

There are several ways in which an attacker can benefit from a relay attack. It should be noted that these are simply worst-case scenarios to illustrate how relay attacks might be used to circumvent security measures. These scenarios are not necessarily practical and could be prevented by current security mechanisms. I mainly use payment systems as an example for the purpose of discussing these scenarios, so I will first explain some basic terminology. The *merchant* is in possession of a reader provided by the *acquirer*, i.e. the bank that asks for the money, which verifies the token on his behalf. The *holder* is in possession of a token and uses it to gain services from a merchant. The token is provided by an *issuer*, i.e. the holder's bank. The payment scheme is controlled by the *operator*, e.g. Visa, that looks after the interaction between all the different entities. In a simple payment system, for example, the issuer gives the holder a token. When the holder wishes to purchase a service from the merchant, he presents his token to the reader. The reader verifies the token, either off-line or by consulting another entity, before notifying the merchant that he can supply the service. The operator subsequently ensures that the transaction is settled between the issuer and the acquirer.

In general, a relay attack is seen as an attack by a fraudulent third party against an honest merchant and holder. In this scenario, the attacker can masquerade as the real holder, so he can circumvent the security of several payment and access control systems. An attack against RFID-based credit cards can be implemented in a similar way as the attack in [45], the difference being that the holder is not required to hand over his token to the attacker. An attacker who wants to gain entry to a building simply identifies a holder, possibly out to lunch, who has a legitimate token. The attacker then activates and interacts with the legitimate token, which allows the attacker's accomplice, holding the proxy-token, to open the required door. The relay attack might, however, also be used in scenarios 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. His accomplice then wanders around outside with the proxy-reader and saps money from tokens. 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 issuer or operator cannot easily distinguish this attack from the regular activity on the merchant's account. 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.

There have been proposals for dual purpose RFID tokens, which contain different payment systems. 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 such systems, a fraudulent merchant could possibly set up a fake top-up reader to acts 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 the proxy-reader relaying information to the debit card reader. A person wishing to top-up his travel credit first enters the amount he wishes to add. After payment he then briefly touches 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. To implement a suitable proxy-reader for this scenario is difficult, as it would need to modulate the forward data onto the RF carrier of the real reader. It would not be able to modulate only its own carrier since the real reader's carrier will cause interference, e.g. the token will always receive a carrier even if the proxy-reader stops transmitting. As a result the proxy-reader would need to cancel the genuine carrier by transmitting a 180° phase shifted version to achieve 100% ASK modulation. This attack could possibly be detected if the travel and payment systems look for simultaneous transactions.

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 possibly allow the attacker to exploit further vulnerabilities in the security protocols of the RFID system. The relay attack is not without limitations. Unless there are vulnerabilities in the security protocol, an attacker cannot modify the data he relays without being detected. A relay attack, therefore, has limited success against systems that require additional verification of the holder, or implement 'two factor' authentication. An attacker, for example, would struggle to execute the attack against RFID-enabled passports if the photo read from his 'passport' does not resemble him. The attacker must also 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 the proxy-reader is within range of a suitable token.

4.3 Practical implementation

The theory of a relay attack is quite simple but implementing the attack is still a practical challenge. In this section I describe a practical implementation of a relay attack against ISO 14443A RFID systems. My goal was to illustrate that it is feasible for an attacker to build a cheap proxy-token and proxy-reader that is capable of attacking an actual RFID system.

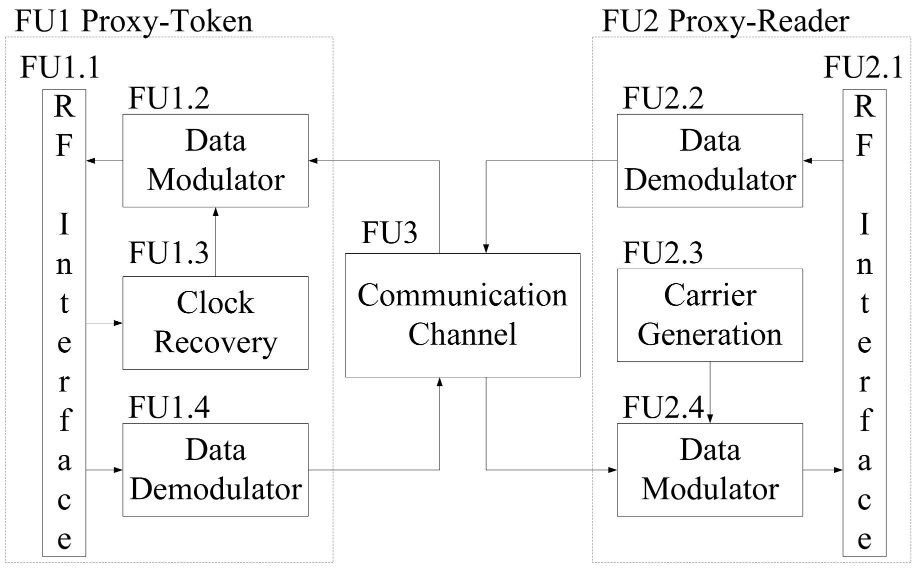


Figure 4.2: Functional diagram of the relay system

4.3.1 Design specifications

A relay attack system, as already described in Section 4.2, consists of three main parts: a proxy-token, a proxy-reader and a communication channel. A block diagram outlining the tasks and interactions of each part, or Functional Unit (FU), is shown in Figure 4.2.

FU1 – Proxy-token

The proxy-token communicates with the reader and functions in the same way as a normal token. The attacker does not need to extend the token's operating range as the device can be held in close proximity to the reader. In fact, the person executing the attack needs to appear as normal as possible and will probably be holding a wallet or bag against the reader. The RF interface for a proxy-token can therefore be identical to that of a real token apart from the fact that the proxy-token can have its own independent power supply. The RF interface (FU1.1) is coupled with the reader's antenna from which it receives a 13.56 MHz carrier signal. The carrier is modulated by the reader for reader-to-token communication (forward channel) and by the token for token-to-reader communication (backward channel). The antenna's resonant circuit should be tuned to 13.56 MHz and have a sufficient Q value to allow both the forward and backward channel data to be transmitted. The data modulator (FU1.2) modulates the backward channel data onto the carrier using load modulation. The modulator needs to switch the impedance of the token, thereby changing the amplitude of the carrier according to the 106 kbit/s Manchester encoded data it is provided with. The modulator also requires a 847 kHz sub-carrier, which is provided by the clock recovery section (FU1.3). The data demodulator (FU1.4) recovers the forward channel data. It should therefore amplitude demodulate the incoming signal and output 106 kbit/s Modified Miller encoded data.

FU2 – Proxy-reader

The proxy-reader communicates with the token and should act like a normal reader. 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 and the Q factor of the antenna used [152,161]. Ideally, the attacker would try to extend the operating range to avoid detection, so the attacker might implement a skimming attack as already described in Chapter 3. The RF interface (FU2.1) should therefore be designed for 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 can be built into a briefcase, clothes, a fake racquet, etc. The data demodulator (FU2.2) recovers the load modulated side-band data located at 13.56 MHz $\pm$ 847 kHz and outputs 106 kps Manchester encoded data. The data modulator (FU2.4) amplitude modulates the 106 kbit/s Modified Miller encoded data onto the generated 13.56 MHz carrier (FU2.3) with a modulation index of 100%.

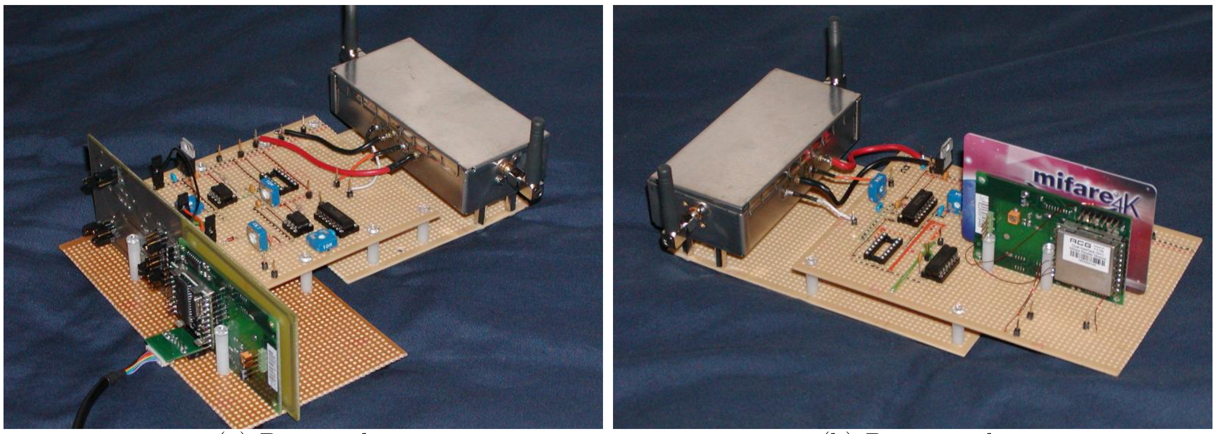
FU3 – Communication channel

The communication channel relays data between the proxy-reader and proxy-token. The channel receives Modified Miller encoded data as input from the proxy token and outputs Modified Miller encoded data to the proxy-reader. Similarly, the channel receives Manchester encoded data as input from the proxy-reader and outputs Manchester encoded data to the proxy-token. The channel itself can alter the data in any way, i.e. stretching or delaying pulses, and transmit data in any format. The communication channel potentially causes the largest time delay in the system. It is therefore important that the channel is designed to relay the data over the required distance while staying within any time limits imposed by the RFID system.

ISO 14443A timing requirements

Relaying data causes a delay, so it is important to consider the timing constraints. The ISO 14443A standard specifies a number of timing requirements for communication.

ISO 14443, Part 3: The reader periodically polls for new tokens using the *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 also be able to respond to the *REQA* command with an *ATQA* within 5 ms after first receiving an unmodulated carrier. This requirement 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. An attacker's response would however need to adhere to the 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. The token will only respond at those times, since it thinks that it is speaking to a real reader, which means that the relay process will have to be very quick or that the attacker will have to get some information, such as the values of the token's *ATQA* and UID in advance.
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* 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.



(a) Proxy-token (b) Proxy-reader

Figure 4.3: Implemented hardware

4.3.2 Experimental hardware

I implemented a proxy-token and a proxy-reader suitable for executing a relay attack against ISO 14443A tokens. 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 £100, with most of the cost being an OEM RFID reader for the proxy-reader.

The proxy-token and proxy-reader hardware is shown in Figure 4.3. I was not concerned with making the prototypes compact, or covert, as they were meant to be proof-of-concept hardware. Size can be reduced by designing a PCB and using surface mount components. The hardware described in this section is not the only possible implementation of a relay attack. An attacker can choose a number of different approaches to implement an attack depending on his skill and resources. For example, Kasper also described hardware capable of performing a relay attack [88]. If an attacker does not have the engineering skills to build his own hardware he could possibly use existing NFC devices when they become available. Alternatively, he could use an open source reader and token such as the OpenPCD and OpenPICC currently being developed [126].

Proxy-token

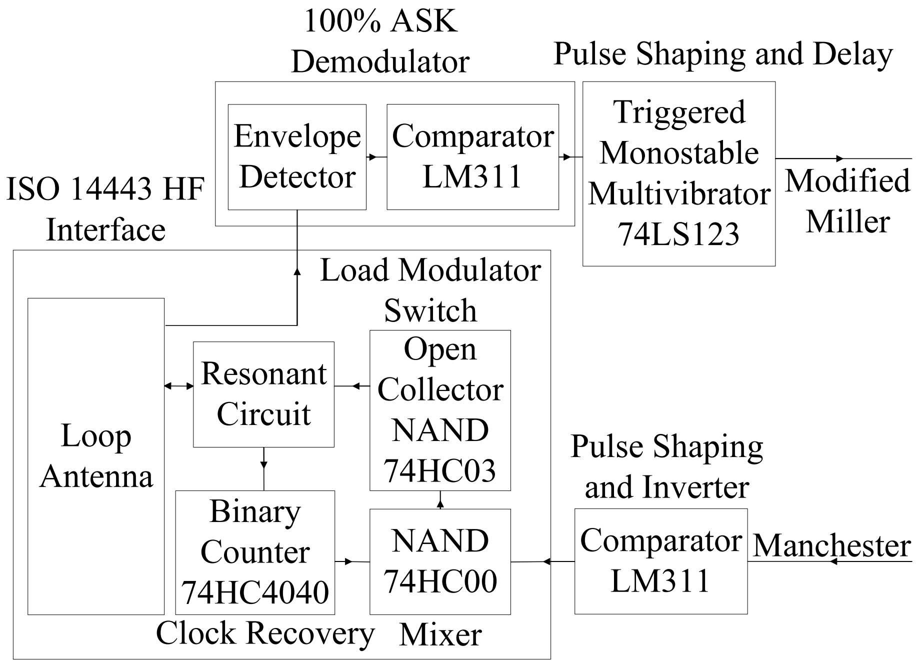


Figure 4.4: Hardware diagram of the proxy-token

The proxy-token, as shown in Figure 4.4, contains an ISO 14443 HF interface, an ASK demodulator and some signal shaping functions. The HF interface is based on a circuit design described in [54, pp 276–278]. Clock recovery is done by using a binary counter (74HC4040) that divides the carrier signal frequency by powers of 2, i.e. 2, 4, 8, 16, . . . , 4096. This clock recovery method does not provide a stable clock for receiving the ISO 14443A forward channel data, since the counter only increments when a carrier is present, which is not always the case when using 100% modulation. This is, however, not a problem because the clock recovery is only required when transmitting the backward channel. The generated 847 kHz subcarrier ($f_{\text{carrier}}/16$) is "mixed" with the returning 106 kbit/s Manchester encoded data using a NAND gate (74HC00). The combined signal is then used to control an open collector output inverter (74HC03) that switches on an additional resistive or capacitive load. This alters the impedance of the resonating circuit and as a result the carrier's amplitude is varied. The Modified Miller data sent by the reader is recovered by an amplitude demodulator. The signal on the proxy-token's antenna is passed through an envelope detector consisting of a diode to rectify the signal and a low-pass filter to remove the HF carrier. The detector's output is then converted to a two-level digital signal using a comparator.

In order to interface to the communication channels the proxy-token also does some pulse shaping of the Modified Miller and the Manchester encoded data. To allow for timing adjustments an adjustable delay is also provided. The delay is implemented with a triggered monostable multivibrator (74LS123), which generates a pulse on the rising, or falling, edge of its input. The width of the pulse can be set by varying a RC circuit and can be either positive or negative polarity. A delay can therefore be implemented by making the device generate a positive pulse of width t_{delay} on the falling edge of the Modified Miller pulse. Another device then generates a 3 μs Modified Miller pulse on the falling edge of the delayed pulse. The Modified Miller pulse is now effectively delayed by t_{delay}. Pulse shaping is described later when discussing the communication channels.

It should be noted that similar tokens adhering to the ISO 14443B and ISO 15693 standards can also be constructed. The circuit design in [54] allows for BPSK modulation of the 847 kHz sub-carrier as needed for the backward channel of ISO 14443B. The clock recovery method also generates a 423 kHz ($f_{\text{carrier}}/32$) sub-carrier suitable for the modulation of the ISO 15693 backward channel data. In each case the load modulation method and the amplitude demodulation of the carrier to recover the forward channel data remains the same.

Proxy-reader

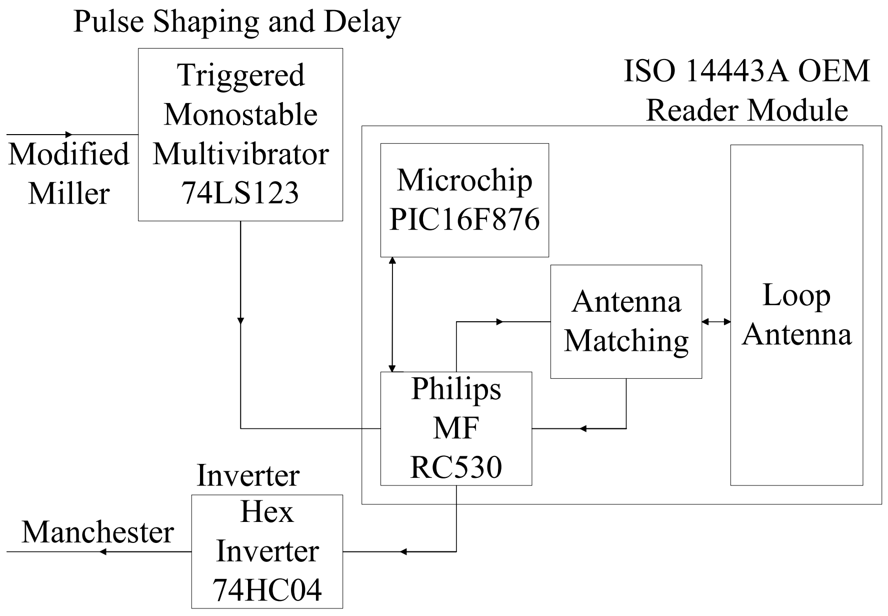


Figure 4.5: Hardware diagram of the proxy-reader

The proxy-reader as shown in Figure 4.5 contains an existing RFID reader and some signal shaping functions. The RFID reader is designed around the NXP MF RC530 contactless reader IC that is compatible with the ISO 14443A standard. The reader IC is controlled by a Microchip PIC16F876 8-bit microcontroller, which is in-circuit programmable. This RFID reader is therefore an ideal hardware development platform, since no new hardware needs to be implemented and the functioning of the reader IC can easily be reconfigured by reprogramming the microcontroller. The MF RC530 IC generates a 13.56 MHz carrier and performs all required modulation and demodulation of data communicated via the contactless interface. The reader IC can be configured to act as an HF interface only. In this configuration, the IC modulates the signal on its *MFIN* pin onto the carrier using 100% amplitude modulation. The Modified Miller data can therefore be modulated onto the carrier by connecting the input data stream to the *MFIN* pin. The IC also recovers the side-band data of the backward channel and outputs the demodulated data on its *MFOUT* pin. The Manchester encoded data stream is therefore provided on the IC's *MFOUT* pin. In order to interface to the communication channels the proxy-reader also performs some pulse shaping of the Modified Miller and the Manchester encoded data. Pulse shaping is described later when discussing the communication channels.

Alternatively, an attacker could build his own HF interface or decode the data and use the reader IC in its normal configuration. An attacker, for example, can decode the Modified Miller data and instruct the reader IC via its serial or parallel communication interface to transmit the data. The reader IC will also demodulate and decode the token's response and send it back to the attacker. The attacker then encodes the data again before sending it to the proxy-token. NXP makes similar contactless reader IC's that are also compatible with ISO 14443B (MF RC531) and ISO 15693 (MF RC632) standards. As far as I am aware, the debug HF interface configuration using the *MFIN* and *MFOUT* pins only supports ISO 14443A data. An attacker would therefore have to implement the alternative proxy-token design if he wishes to implement a relay attack against tokens adhering to the other two HF standards.

Communication channel

The communication channel relays the digital Modified Miller and Manchester encoded data using short range RF communication. I obtained two ISM-band RF channels, that used Micrel's QwikRadio UHF ASK/OOK integrated transmitter and receiver ICs, with a maximum range of approximately 50 m. These ICs have a nominal filter bandwidth that is sufficient for data throughput rates of up to 115 kbit/s. The RF modules are built into metal cases to try and minimize high-frequency interference from the readers' antennas. 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 3 μs Modified-Miller pulses exceed the bandwidth of the RF channel's filter, so the proxy-token decreases the required bandwidth by stretching the pulses. The stretched pulses received are reshaped to their original width by the proxy-reader. To prevent the automatic gain control from amplifying background noise when the carrier is keyed off for a length of time both channels transmit a '1', logic high, when idle. A '0', logic low, only then occurs during data transfer. The Modified-Miller code is already in this format but it is necessary for the proxy-reader to invert the Manchester data. The proxy-token uses an inverting comparator to invert the data again in addition to sharpening the pulse edges deformed by the RF filtering. The bandwidth of the Manchester and Modified-Miller encoded data is greater than that of the RF-channel, so the transmitter and receiver ICs were operated beyond their normal specifications. This was not a problem when relaying shorter commands, like *REQA* and *SELECT*, although bit-errors sometimes occurred during longer sequences. These channels, however, still served their purpose, which was to illustrate that it is feasible to relay data within the time constraints imposed by the

ISO standard. An attacker wishing to construct a relay attack, on a bigger budget, would be able to obtain similar RF receivers with sufficient channel bandwidth.

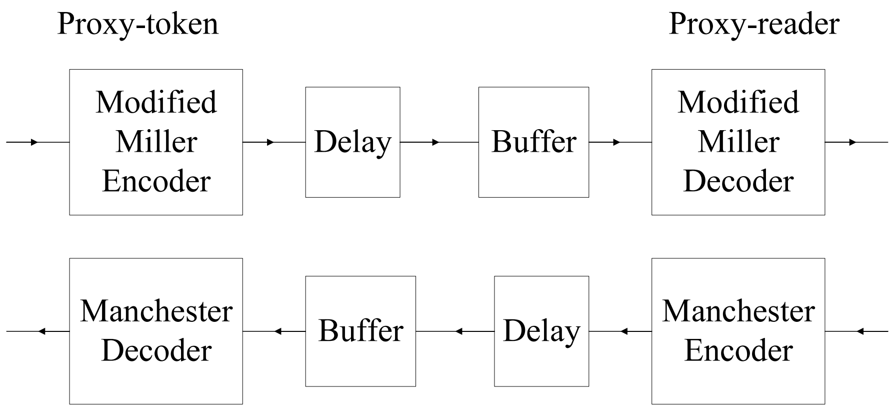
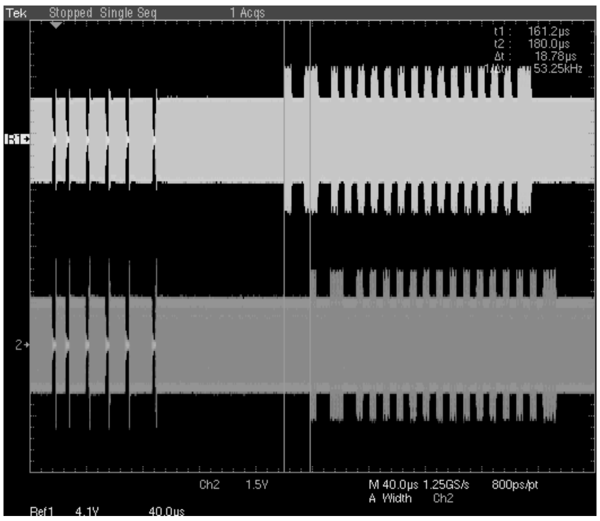


Figure 4.6: Data flow within the FPGA ‘communication channel’

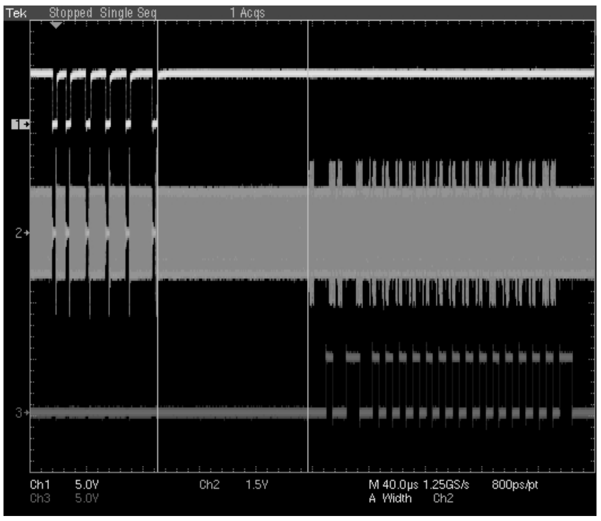
My hardware design initially aimed to minimize the attack delay, based on early assumptions that timing constraints might be strictly enforced. Once the attack succeeded, and the question arose of how long the attack delay could be before being detected, this RF channel was no longer that useful as an investigative tool. I therefore simulated a relay channel using an FPGA development board connected between the proxies as shown in Figure 4.6. The FPGA stores the decoded Modified Miller sequence received from the proxy-token in a FIFO buffer. After a delay, the FPGA again encodes the stored data and outputs a Modified Miller sequence to the proxy-reader. The delay starts when the first bit of the sequence is buffered and can be set to multiples of the bit period, $k{\cdot}9.44$ µs with $k > 2$. The FPGA similarly stores the decoded Manchester sequence from the proxy-reader in a buffer. After another delay the data is provided as output to the proxy-token as a Manchester encoded sequence. Since the delay can be changed this ‘channel’ can be used to determine how much delay a communication channel can introduce before the RFID system prevents the attack. This setup also allowed for modification of the buffered data before being encoded again.

4.3.3 Results

The relay attack was successfully implemented against an ACG Multi-ISO reader and a NXP Mifare Classic token using the proxy-token and proxy-reader with the RF link. I used the ACG reader and NXP token as they were a good examples of a generic ISO 14443A system. I do not wish to imply that these products are more at risk of a passive relay attack than any other products that might not have been used. The short range RF communication channel was used because I assumed that the attacker would need to minimize the delay in order to avoid detection. However, the timing constraints as defined in the standards were not strictly enforced and there was sufficient time allowed to relay messages. According to the standard, the reader should only accept responses to the *REQA* and *SELECT* commands if they adhere to a FDT where $n = 9$. The reader I used, however, accepted the proxy-token’s response even though the value of n was effectively 11 as a result of the additional relay delay of approximately 19 µs shown in Figure 4.7(a). The majority of the delay was introduced by the RF link and the backward channel demodulation process of the RFID reader in the proxy-reader. Figure 4.7(b) shows the input on the *MFIN* pin and the output on the *MFOUT* pin relative to the reader's carrier. There is a negligible delay between the input of the Modified Miller data and the corresponding 100% amplitude modulation. The Manchester output is, however, delayed by 9.44 µs, which is equal to a single bit period.



(a) Timing comparison between a token (top) and proxy-token (bottom). The tokens send an *ATQA* in response to a *REQA* command



(b) Example of a relayed data sequence at the proxy-reader. Modified Miller *REQA* command (top), carrier modulation (middle) and Manchester *ATQA* response (bottom)

Figure 4.7: Delay introduced by the relay hardware

Even though the value of n did not seem to matter, the reader's receiver did expect the token's response to adhere to the general FDT bit-grid as described in Section 4.3.1. 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. This is to be expected if one considers the receiver structure implemented by the reader. The backward channel is recovered using a correlation receiver. After the carrier is removed, the Manchester data sequence is correlated with the base function, which is a 4.72 µs square pulse. The receiver effectively integrates the data sequence over time periods equal to half the bit period. The receiver then samples the result in the middle and at the end of the bit period. This allows the receiver to evaluate the value of the first and second half of the bit period to determine whether a '1' or '0' was transmitted. The sampling clock is generated by the receiver to correspond to the bit-grid defined by FDT. A response that is phase shifted relative to the sampling clock will be sampled at the incorrect time intervals, causing the data to be evaluated incorrectly. The allowed phase shift is dependent on the evaluation threshold, e.g. if a response is delayed slightly but the integration result of the high bit-half is still above the threshold when sampled, the bit's value will be evaluated correctly. I used the simulated relay channel to increase the relay delay and found that the reader continued to accept any response as long as the data was aligned to the required bit grid. This conclusion was also drawn by Kasper [88], who determined experimentally that a reader implemented with this contactless IC accepted responses starting during a time slice of 2.5 µs every 9.44 µs.

The attacker only encounters a possible problem when he participates in the anti-collision procedure together with a real token. Since the real token adheres to the $n = 9$ condition, a delayed response would be detected, as the total length of the response would be greater than expected. For example, in the case where $n = 11$ the reader will receive two extra bit periods of data since the relayed response starts two bit periods after the real token's response. Alternatively, the misaligned bits will be interpreted by the reader as collisions and it will be unable to select a token. Within the context of our attack this scenario should not occur as the attacker's proxy should be the only 'token' interacting with the reader.

Even though low-level timing constraints were not enforced, the attacker is limited by higher layer timeouts. The RFID tokens I had available for testing did not implement ISO 14443-4 and therefore the FWT timeout was not implemented. The reader's configuration software did, however, 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. Using the simulated relay channel I verified that the reader did implement the default 4.8 ms timeout for the *REQA* command. Although the implemented timeouts complicate the relay attack by placing constraints on the hardware it does not prevent the attack. Both my attack hardware and the setup discussed in [88] easily implemented the attack in under 300 µs. I therefore believe that it is feasible for an attacker with the necessary resources to implement a relay attack that is completed within 4.8 ms. The attacker can also circumvent the timeouts for the *REQA* and *SELECT* commands by getting the token's *ATQA* and *SAK* responses along with its UID. These values can then be stored in the proxy-token and sent to the reader when required without any delay [88]. In this attack scenario the attacker would be able to execute the relay attack even if the reader enforced the $n = 9$ FDT condition. Specified timeouts and timing constraints defined in the ISO standard do therefore not provide adequate protection against relay attacks.

I also considered the possibility that an attacker could alter data before relaying it back to the reader. Using the simulated channel setup I successfully modified both the forward and the backward channel data. The active attack did not require any additional delay compared to the passive attack implemented using the same channel. The ability to modify data allows the attacker additional attack options. 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. The attacker might also be able to exploit weaknesses in the security protocols implemented by the RFID system, as described in the next section.

4.4 Security implications

A relay attack circumvents most application layer security mechanisms. To be successful, the attacker only needs to relay the communication between the token and the reader for the duration of the transaction. The success of the attacker does therefore not rely on a weakness in a specific protocol or algorithm. In other words, a high security token implementing AES with a 256-bit key and a medium security token implementing a proprietary cipher with a 32-bit key are equally at risk of this attack. To my knowledge the only cryptographic mechanism for preventing relay attacks are distance-bounding protocols, as discussed in Chapter 5, which will detect the additional delay introduced by the attacker. These protocols are, however, not implemented in current systems. The attack can be made more difficult by implementing 'two factor' authentication, although this cannot prevent the attack entirely.

The relay attack hardware is also an ideal platform for executing a 'real-time' man-in-the-middle attack. During this active relay attack the adversary exploits 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 μ-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.

To get an idea of how a real-world system operates I used my relay hardware to observe a closed payment system implemented at my College to charge students for photocopies. Once my token was inserted the following communication sequence took place, at the time shown, before the reader displayed my current balance:

0 ms	*REQA* command and response
3 ms	*GET* command and response
44 ms	*SELECT* command and response
82 ms	Failed authentication
99 ms	*REQA* command and response
142 ms	*SELECT* command and response
162 ms	Successful authentication
179 ms	Read Data

It is interesting to note that the sequence was the same each time the experiment was repeated. The failed authentication might indicate that the reader supports more than one application. Once it fails to authenticate the first time it realizes that the token does not support the associated application and it tries the next option. It was easy to identify each command even though some frames are encrypted. This is possible because each command has a recognizable format, which can be obtained from the token's data sheet or example code provided by the token manufacturer. 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. As a result a system that does not ensure freshness when encrypting the plaintext data could fall prey to cut-and-splice attacks.

Some tokens only implement a single cryptographic algorithm, which is used to provide both authentication and encryption. Integrity checking is then implemented by error detection mechanisms such as parity checking and Cyclic Redundancy Codes (CRC). If such a system implements a stream cipher it could be vulnerable to an active relay attack because of a combined weakness in the encryption and linear integrity mechanisms. The data is encrypted using a stream cipher, which is linear, so 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. Mechanisms to detect transmission errors are not sufficient for preventing data tampering, i.e. CRC and parity bits can simply be modified in the same way to match the new data since these are linear operators. This is not a new attack and has already been used against other protocols, such as WEP [12]. Figure 4.8 shows how the transmitted data frame is constructed and Table 4.1 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 4.1: Modifying an encrypted value, with odd parity checking, from ‘4’ to ‘132’.

To execute this attack an attacker would require some knowledge about the frame format, the parity bit and the CRC polynomial. The attacker also 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. For example, in the photocopier example mentioned before, I could have flipped bits in the response to the read command to determine which bit positions affect the balance. In reality, an attacker probably needs to modify only one or two bits to change the date his season ticket expires or 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.

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 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 mask. The attacker then flips all the bits in the relayed frame that corresponds to a ‘1’ in the 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 [112, 113]. 3-pass authentication and encryption are implemented with a proprietary stream cipher, Crypto1, that uses a 48-bit key and integrity checking is provided by a CRC, parity bits and bit counting [124, 125]. No further information is given about the format of the 3-pass authentication protocol, except that it adheres to

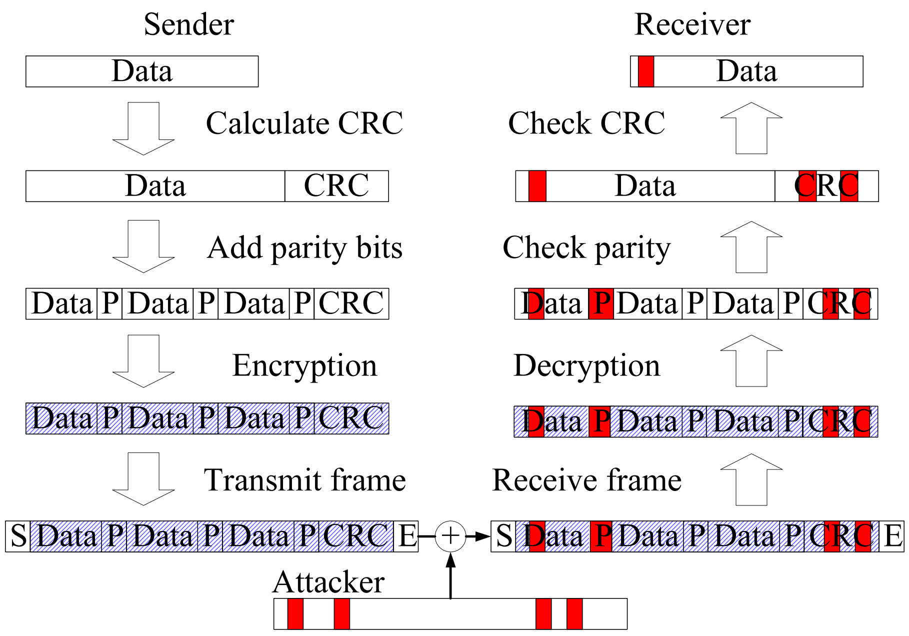


Figure 4.8: Modifying data bits between the token and reader

ISO 9798-2, and the operation of the Crypto1 cipher. Since I already used this token as a generic 14443A token during my experiments I was interested to see whether an active relay attack would also work. I therefore implemented the bit-flipping attack against my test system and successfully managed to modify data during both read and write commands. The reader accepted the modified data frame it received from the proxy-token and the modified value was displayed by the ACG Reader Utility. The token also accepted the modified data frame it received from the proxy-reader and stored the modified value in memory. Further details about the implementation of this attack is withheld as this vulnerability affects a widely deployed product. Even though the token exhibits a vulnerability it does not necessarily mean that every system using these tokens is vulnerable. Systems can be designed to provide additional measures to prevent bit modification. Possible solutions would be to calculate a MAC and store it on the token along with the data, i.e. data$||$MAC$_K$(data), or to encrypt the data with a block-cipher and to store resultant ciphertext on the token. I therefore do not wish to comment further on the possible implications of this attack without further information about the relevant systems.

A final scenario that I wish to mention is the use of relay attacks to assist in tamper attacks. Some low-cost tokens that I studied did not implement any noticeable tamper resistance. An attacker with the necessary resources could therefore recover the plaintext, or key material, by performing invasive tamper attacks on the token [93]. Figure 4.9 shows sections of a decapsulated token that contain possible probings points, including eight pads located next to the EEPROM control logic that could be connected to data bus lines. Under normal circumstances the attacker would not be able to gain much information from probing these points since he cannot authenticate with the token in order to instruct it to retrieve data from memory. With the relay attack, however, he could communicate with the token using the proxy-reader and possibly observe the plaintext data that is exchanged. HF RFID tokens have also been shown to be vulnerable to the same Differential Power Analysis (DPA) as contactless cards [72]. To accurately measure the power consumption of the token the attacker needs special hardware, e.g. a Helmholtz assembly. It is unlikely that an attacker would be able to present his token to a real reader

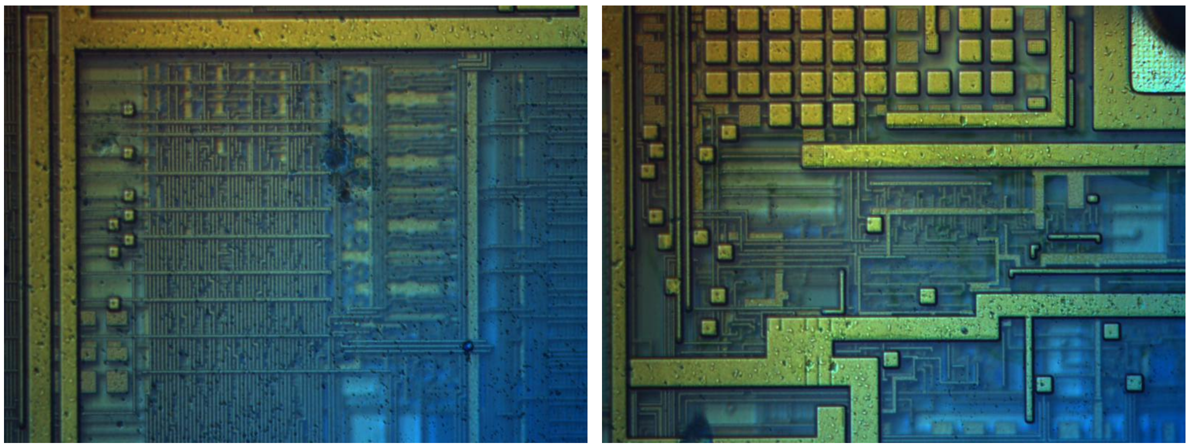

(a) Vicinity of EEPROM (b) Top Right

Figure 4.9: Sections of a decapsulated token showing possible test pads

while covertly recording a power analysis trace. Instead he could use a relay attack setup to make the reader talk to a token being measured remotely.

4.5 Conclusion

RFID devices are often used for proximity identification because the physical limitations of the communication channel implies that the token is in close proximity to the reader. RFID devices are, however, vulnerable to a relay attack, which effectively allows an attacker to 'clone' the victim's token for a short period. It is possible that the legitimate owner will remain unaware of the attack since the attacker can activate the token without his knowledge. It is not only third-party attackers that could benefit from this attack since the holder of the token, or the merchant controlling the reader, could also benefit from executing a relay attack in some cases. Simply relaying information between the token and the reader does not require the same technical resources from the attacker as hardware tampering or cryptanalysis. This attack is therefore a feasible method for circumventing current security protocols with little effort. The attack can also be given an active twist by selectively modifying the relayed communication.

I implemented a practical relay attack against an ISO 14443A RFID system using self-built hardware. The necessary hardware parts were easily obtained and the cost of the whole system could be less than £100, with most of the cost being the OEM reader and RF link. Initially, it was thought that the communication requirements specified in the ISO standard would impose time constraints on the attacker. This would limit the amount of delay his relay hardware can introduce and therefore add to the complexity of the attack implementation. However, lower layer timing requirements were not strictly enforced and communication timeouts allowed enough time to implement the attack. Even if the lower level timing was to be enforced, an attacker could still execute the attack by obtaining the timing-sensitive responses from the token beforehand and storing them in the proxy-token before starting to communicate with the reader. I also discuss how the attack implementation could possibly be used to facilitate further attacks against low and medium security tokens often used in transport payment and identification systems.

For example, an attacker could use the relay hardware to modify transaction data, if the token's security protocol allows this, or as an aid in tamper attacks.

The relay attack is difficult to defend against, as it is not strictly a failure of cryptographic protocols. Physical shielding of the device from radio signals by enclosing it in a Faraday cage consisting of a metal mesh or foil could prevent access by unauthorized parties. This is a viable option in some cases and could prevent unauthorised access to the card. The problem is that the token still has to be taken out of the shield to access services, which provides the attacker with a window of opportunity. An attacker can also try to convince the owner to willingly present his token to the proxy-reader. This relay attack is invisible to application layer security and therefore new protection measures should focus on the physical layer. An attacker wishing to forward data between a token and a reader that are a distance apart will be unable to avoid causing a delay in the system. Distance bounding protocols would therefore be an effective way of preventing relay attacks, although the practical implementation for a low-cost passive token with limited resources remains a challenge. The effectiveness of this attack, especially in the RFID environment, was the motivation behind my research on distance bounding protocols for RFID systems covered in the following chapters.

Chapter 5

Distance bounding: Proof of proximity

Distance-bounding protocols provide assurance as to the distance between two devices. To my knowledge this is the only cryptographic way of preventing relay attacks. Since Brands and Chaum first described distance-bounding protocols in 1993, several new protocols have been proposed. In this chapter I look at the merits and weaknesses of these protocols. I also propose a new distance-bounding protocol for the RFID environment that is tolerant to bit errors, and has simplified setup and verification stages to allow for fast execution.

5.1 Introduction

Distance and location measurements have countless applications, such as use in navigation and construction. Physical location also provides a measure of trust with regards to security. In some systems the users are granted privileges, or services, based on their proximity or location. Verifying the location of a device, through the use of secure protocols, has therefore become important in pervasive environments [22]. Ensuring proximity can enhance traditional authentication mechanisms [19] and can provide additional assurance, such as a metric for secure routing in ad-hoc networks [20].

In this chapter I discuss how to verify the proximity between two devices cryptographically, without the help of additional devices. Brands and Chaum were the first to address this problem by introducing distance-bounding protocols [13]. Distance-bounding protocols determine an upper bound for the physical distance between two communicating parties. This distance can then be used as a secure measure of proximity. In Chapter 4 I showed how an attacker can exploit assumptions about the location-limited nature of the communication channel to perform a relay attack between two devices. Secure distance-bounding protocols are integrated into the underlying communication channel and are meant to detect any extra delay in the prover's expected response. Distance-bounding protocols, if implemented correctly, can be an effective way to prevent relay attacks. Device proximity is also useful information for mapping the topology of the network and for geographically aware routing algorithms [89]. Distance-bounding has therefore also been proposed as a protective measure for wireless networks, where relay attacks (known in this context as wormholes [71]) could be used to circumvent key establishment and routing protocols [69, 70, 87].

In the last few years, there have been a number of proposals for new distance-bounding protocols. I discuss the merits and weaknesses of these further in Section 5.4. Even if the cryptographic part of the proposal is seemingly secure, several proposals are still vulnerable because of the way the communication channel is implemented. Communication channels not optimized for minimal latency imperil the security of distance-bounding protocols, as the attacker can still find ways to commit distance fraud or execute a relay attack. It is also shown in Section 5.4.4 that some proposals are vulnerable to a guessing attack where the malicious prover preemptively transmits guessed values for a number of response bits. In most cases, proposals are also not tolerant to bit-errors in the communication channel. In Section 5.5 I describe a new distance-bounding protocol for the RFID environment that is tolerant to bit errors and has simplified setup and verification stages. This chapter is partly based on a paper I co-authored with Clulow, Kuhn and Moore in 2006, "So Near and Yet So Far: Distance-Bounding Attacks in Wireless Networks" [35]. The guessing attack on single multi-bit distance-bounding protocols, the exploitation of communication latency and the principles for implementing secure ToF (Time-of-Flight) distance bounding were discovered independently by myself and some of the co-authors. This chapter also includes work from a paper I co-authored with Kuhn in 2005, "An RFID Distance-Bounding Protocol" [64]. The proposed distance-bounding protocol for RFID and the weakness of distance-bounding protocols against bit-errors was initially suggested by my co-author, after which we worked together to develop these ideas further.

5.2 Proving your proximity

The communication medium itself has been used to create location-limited, or constrained channels, to provide context information about users [91]. These channels use short range RF [31], directional IR [5, 108, 166] or even contact channels [153] to exchange data. The idea is that two devices can only communicate if in very close proximity, or line-of-sight, of one another. Location-constrained channels prove the device's location implicitly but they fail to prevent relay attacks since an attacker can still use a proxy device to fool the system.

Several proposals construct unforgeable channels to prevent relay attacks by a malicious third party. It is suggested that channel-hopping radio is difficult to track and thus difficult to relay [2]. Alternatively, each device could have a distinguishable fingerprint based on the characteristics of its communication channel [131] so that the verifier can determine who the sender is. These methods do not provide any accurate proximity information apart from 'in communication range', and could possibly be vulnerable against a fraudulent prover. For example, a fraudulent prover knows the channel-hopping sequence so now the verifier is making a distance estimate based solely on the expected transmission range of the prover, which is a parameter the prover could change by amplifying the transmitted signal. Further proposals add hidden information to the location data [69, 98]. The first method requires the verifier to know its location, which requires extra information from other devices and disqualifies it for two-party distance-bounding. The second method was suggested for GPS where the sender is trusted and therefore it does not protect against a fraudulent prover.

Secure location services provide relative or absolute location of devices within a specific network [6,169]. A device within the network not only estimates the distance to another device but also collaborates with other devices, including 'anchor' or base stations that provide trusted reference locations [86]. The verifier can therefore be assisted by other devices to cross reference, repeat and verify measurements to defend against malicious behavior [21,22,102,103]. Secure location protocols protect against relaying attacks and fraudulent senders.

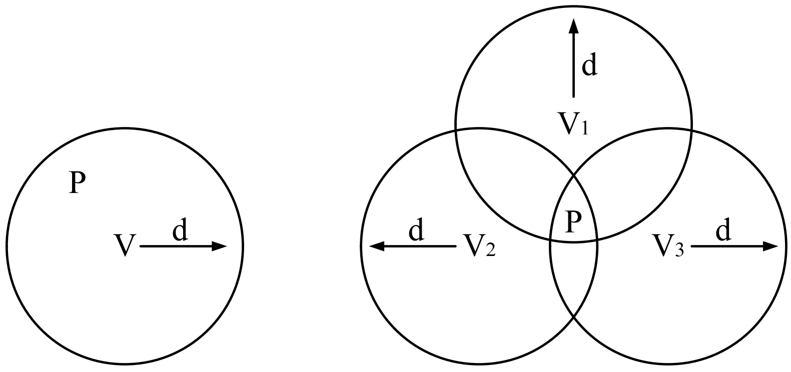


Figure 5.1: Distance-bounding vs location

In some scenarios two devices need to establish trust without a network of trusted devices providing assistance. Distance-bounding protocols only aim to prove the relative proximity of two devices. As shown in Figure 5.1, distance bounding only involves two parties, a prover and a verifier, and allows the verifier to place an upper bound on the physical distance to the prover. Distance-bounding protocols are, however, used as building blocks in secure localization or positioning proposals. The protocol should be resistant to attacks from a fraudulent prover and a malicious third-party. Unlike secure location services where the verifier can get extra information from other trusted devices, the distance bounding verifier relies exclusively on information gained from executing the protocol with the prover. Secure distance-bounding protocols are also integrated into the underlying communication channel, so the security of the protocol not only depends on the cryptographic mechanisms but also on how the physical attributes of the communication channel are used to measure proximity.

Conventional location-finding techniques generally used for proximity measurement are:

- **Received-Signal-Strength (RSS):** Uses the inverse relationship between signal strength and distance to estimate the distance to other nodes.
- **Angle-of-Arrival (AoA):** Examines the direction of received signals to determine the locations of transmitters or receivers.
- **Time-of-Flight (ToF):** Measures elapsed time for a message exchange to estimate the distance based on the communication medium's propagation speed.

Systems have been demonstrated that can estimate location with typical errors as small as 1.5 m, by processing received signal-strength information from multiple base stations [6,23,96]. A trust system for RFID that is based on signal strength has also been proposed [55]. However, RSS and AoA methods are not ideal since attackers can easily alter received signal strength, by either amplifying or attenuating a signal, and angle-of-arrival, by reflecting or retransmitting from a different direction.

This leaves only time-of-flight as a possible mechanism for securely determining proximity and the method most often described in distance-bounding protocols. Both radio frequency (RF) [169] and ultrasound channels [65, 107] have been used in indoor ToF location systems. The propagation speed of sound is much slower than that of light, so it is easier to obtain high spatial resolution using simple hardware. This property, however, makes ultrasound vulnerable to a relay attack where messages are forwarded over a faster communication medium. In contrast, the propagation speed of radio waves in air approaches the speed of light. Thus it resists simple relay attacks since information cannot propagate faster than this. The attacker can only make a device appear further away by blocking a legitimate device's communication and sending a delayed version to the intended receiver.

Although implementation on constrained devices can be a challenge, RF is already an established medium for mobile communication. It is therefore an ideal candidate for implementing distance-bounding systems. In Chapter 6, I discuss the implementation of communication channels for distance-bounding. The rest of this chapter focuses on ToF distance-bounding protocols.

5.3 Time-of-flight distance-bounding

Time-of-flight uses the fact that the maximum propagation speed of the communication medium is constant and known. There are two well known examples of how ToF is generally used to determine distance: GPS and radar. In global positioning systems signals flow only in one direction. The receiver can then estimate its own position if it knows the transmitter's position, or it can estimate the position of the transmitter if it knows its own location. The distance between two devices can be calculated as

$$d = c \cdot t_{\mathrm{p}} \tag{5.1}$$

where c is the propagation speed and t_{p} is the one-way propagation time between the transmitter to the receiver. A distance-bounding protocol using this approach could possibly be implemented as follows: We assume that both the verifier and the prover share a common, high-precision time base, e.g. secure GPS receivers. The verifier sends out a challenge C at time t_0, which the prover receives at time $t_0 + t_{\mathrm{p}}$. The prover then replies with a message-authenticated data packet $\{t_0 + t_{\mathrm{p}}, C'\}_K$, where C' is the challenge he received. The verifier checks the message-authentication code of this packet with the shared key K and also whether $t_{\mathrm{p}} \leq d/c$, where d is the upper bound for the distance and c is the speed of light. An attacker relaying the challenge would introduce extra propagation delay, which would be reflected in t_{p}, and as a result the verifier would conclude that the prover is outside the allowable distance bound. There are, however, some problems with this implementation. An accurate shared timebase is difficult to achieve, especially on resource-constrained devices, so this method is not practically feasible in all cases. This protocol also assumes that the prover is a trusted entity, thus making the prover responsible for taking the time measurement. There are no measures in place to prevent the prover from lying about $t_0 + t_{\mathrm{p}}$ and appearing closer to the verifier as a result. From a security perspective this is not acceptable as a distance-bounding protocol should provide the verifier with reliable proof that the prover is actually within a certain distance, even in cases where the prover is not trusted.

The problem of maintaining a shared timebase in both the verifier and the prover could be prevented by making distance measurements based on Round Trip Time (RTT). In RTT systems, only the verifier is required to maintain an accurate timebase and the prover is also no longer responsible for providing the security critical time measurements. The distance between the prover and verifier can be calculated as

$$d = c \cdot \frac{t_\mathrm{m} - t_\mathrm{d}}{2} \tag{5.2}$$

$$t_\mathrm{m} = 2 \cdot t_\mathrm{p} + t_\mathrm{d} \tag{5.3}$$

where c is the propagation speed, t_p is the one-way propagation time, t_m is the measured total round-trip time and t_d is the prover's processing delay between receiving a challenge and sending a response.

A distance-bounding protocol using RTT could possibly be implemented as follows: The verifier again sends a challenge C at time t_0, which the prover receives at times $t_0 + t_\mathrm{p}$. Once the challenge is received the prover sends a response R in the opposite direction at time $t_1 = t_0 + t_\mathrm{p} + t_\mathrm{d}$, which the verifier receives at time $t_2 = t_0 + 2 \cdot t_\mathrm{p} + t_\mathrm{d}$. The verifier determines $t_\mathrm{m} = t_2 - t_0$, checks that the response is correct and finally confirms that $t_\mathrm{p} \leq d/c$, where d is once again the upper bound for the distance. Although this protocol is an improvement on the first example extra care must be taken to specify the nature of the response. If the prover simply echoed the challenge C $(R = C')$ it would provide limited security as any attacker, who controlled a proxy-token close to the verifier, would be able to do this and achieve an acceptable measurement $t_\mathrm{p} \leq d/c$. Ideally, the response should depend on the challenge, i.e $R = f(C')$, as this is also an effective measure against fraudulent provers. If this was not the case, and R was merely a random nonce agreed on beforehand between the prover and verifier, the prover could send R before receiving C and effectively decrease the round-trip time. Specifying that R is a function of C forces the prover to wait until he receive the challenge and prevents him from preemptively transmitting his response.

Having introduced the most basic protocol design requirements, during which some attacks scenarios were briefly discussed, it is time to provide a more formal description of what the main attacks are that distance-bounding protocols aim to address. There are three classes of attacks, as shown in Figure 5.2, that are presented in literature with regards to distance-bounding protocols:

- **Relay Attack:** A fraudulent third party tries to convince the verifier that the prover is in close proximity. Both the verifier and the prover are honest and unaware of the attack. This was first described as 'mafia fraud' in [43] and is also known as a 'wormhole' attack in sensor-network literature [69].

- **Distance Fraud:** The prover is fraudulent and tries to convince the verifier that he is closer than is actually the case. Distance fraud was first discussed in [13].

- **Terrorist Attack:** The prover collaborates with an attacker, who wants to convince the verifier that the real prover is in close proximity. The prover's motivation could either be that he is fraudulent, or that he is honest but being coerced by the attacker. The "terrorist attack" was first described in [42].

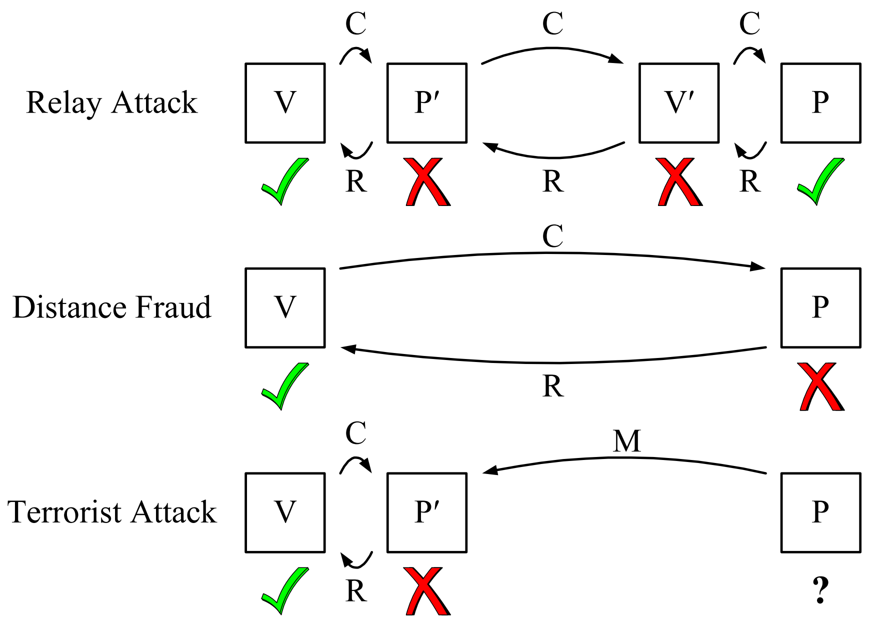


Figure 5.2: Three main attack classes considered for distance-bounding protocols.

In most cases, protocols aim to prevent distance fraud and relay attacks. Some distance-bounding proposals do not consider the "terrorist attack" as it is generally an accepted condition of security that the participants do not reveal their secrets. If the prover reveals his responses, the material M needed to calculate his responses or his key to the attacker, the verifier cannot distinguish between the prover and the attacker. Preventing the "terrorist attack" altogether is therefore all but impossible. Some protocols do, however, force the prover to reveal a valuable secret, e.g. long term key, if it reveals all possible responses to a third party. This approach allows a third party, who manages to coerce a prover into revealing all its responses, to gain the long term secret. Whereas, in the original case, the attacker only acquired the responses to execute a distance bound once, he could now masquerade as the prover multiple times. The attacker could also then use this secret in another environment, e.g. prover shared his information to circumvent a door access control, but now the attacker can also get into his bank account. It is argued that this scenario could deter the prover from cooperating with an attacker.

5.4 Existing protocol proposals

In the last few years, several protocols have been published that allow a verifier to determine an upper-bound on the physical distance to a specific prover. These proposals can further be classified by how they implement different stages of the distance-bounding process. Most of these distance-bounding protocols consist of three basic stages:

- **Setup:** The verifier and prover prepare for the exchange stage.
- **Exchange:** Timed exchange of challenge and response data.
- **Verification:** The verifier ensures that the exchange step has been executed faithfully and can therefore use the RTT to calculate the distance.

In the rest of this section, I discuss several protocols described in the literature. I use the three protocol stages, along with the attacks that they claim to prevent, as the basis of my comparison. It is not possible to provide comprehensive detail about each protocol, so I recommend that the reader consults the original publications for further details.

5.4.1 Timed authentication protocols

These protocols are the simplest form of ToF-based distance-bounding, with the verifier timing normal data exchanges. The basic idea is to execute a challenge-response authentication protocol under a very tight time-out constraint, which was a concept first proposed by Beth and Desmedt [9]. For example, a verifier V transmits a random n-bit nonce $N_V \in_{\mathrm{R}} \{0,1\}^n$ to the prover P, who replies with a message-authentication code $h(K, N_V)$, where h is a keyed pseudo-random function and K is a shared secret key. Numerous protocols have been proposed using different constructions for pseudo-random functions keyed with shared secrets, public-key mechanisms, or trusted third parties. Although all distance-bounding protocols could be seen as 'timed', this set of protocols generally time an exchange without considering variations in the processing time of the response or the format of the challenge and response. This results in possible inaccuracies in the round-trip time measurement. For example, a smart token that usually takes 100 ms to compute a public-key signature and has a 1% (1 ms) processing time variation could cause a 333 km error in the distance estimate. Furthermore, the long exchange strings introduce latency that an attacker can exploit, as shown Section 5.4.4.

Secure neighbour detection (2003)

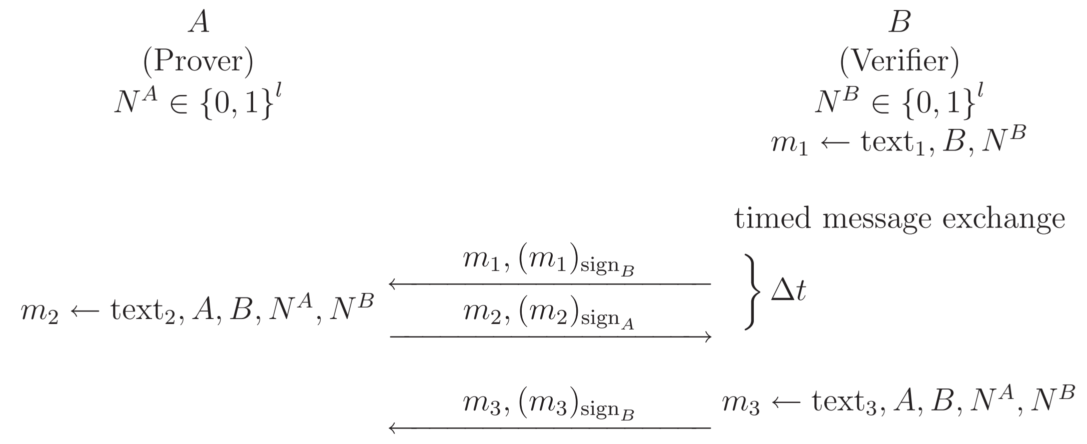

A
(Prover)
$N^A \in \{0,1\}^l$
B
(Verifier)
$N^B \in \{0,1\}^l$
$m_1 \leftarrow \text{text}_1, B, N^B$
timed message exchange
$m_1, (m_1)_{\text{sign}_B}$
Δt
$m_2 \leftarrow \text{text}_2, A, B, N^A, N^B$
$m_2, (m_2)_{\text{sign}_A}$
$m_3, (m_3)_{\text{sign}_B}$
$m_3 \leftarrow \text{text}_3, A, B, N^A, N^B$

Figure 5.3: Secure neighbour detection distance-bounding protocol

The secure neighbour detection protocol proposed by Hu, Perrig and Johnson [70], as shown in Figure 5.3, is an instance of a timed authentication protocol where the elapsed time Δt during the exchange of signed nonces is then used to calculate a distance-bound. In the setup stage the verifier and the prover both generate a random nonce of length l. The verifier also constructs a 'Solicitation' message m_1, which contains data in addition to the verifier's ID and nonce. The verifier signs m_1 and transmits both the signature and the message to the prover. The prover responds with a 'Reply' message m_2 containing data in addition to the prover's ID, the prover's nonce, the verifier's ID and the verifier's nonce. The verifier times the round-trip delay Δt_i between sending m_1 and receiving m_2. Finally, the verifier transmits a 'Verification' message containing data, the verifier's ID, the verifier's nonce, the prover's ID and the prover's nonce.

Theoretically, when only taking the cryptographic mechanisms into account, this protocol prevents distance fraud, relay attacks and terrorist attacks. This is, however, not the case if the practical implementation is taken into consideration. During the exchange stage there are significant processing overheads, such as verifying and signing messages, and the exchanged messages themselves are very long. While the authors discuss mechanisms for increasing the efficiency of the signing operations, the associated delay renders the bound inaccurate and unreliable, i.e. every 100 ns change in the processing time t_d alters the distance-bound by 30 m. A fraudulent prover or third party attacker could exploit the length and variability of the processing delay, e.g. a device with higher performance components can extract a time advantage and commit distance fraud by performing these operations faster. This protocol does not allow for communication errors during the exchange stage.

Echo protocol (2003)

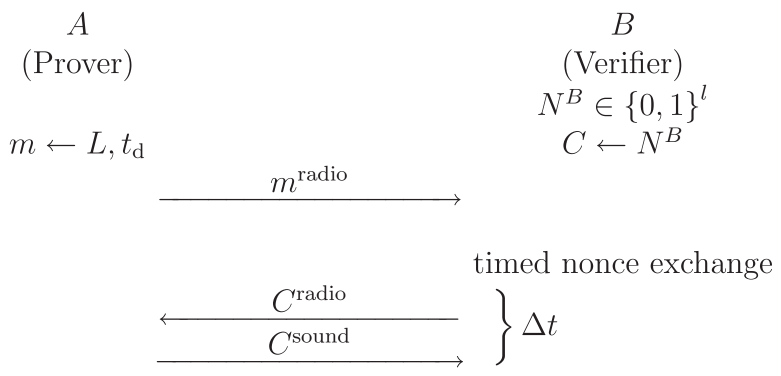


Figure 5.4: Echo distance bounding protocol

Sastry, Shankar and Wagner [145] proposed a dual-medium protocol, as shown in Figure 5.4, to verify a prover's claimed physical location L within a circular region R centered on the verifier. During setup the prover transmits m containing his location claim L and expected processing delay t_d while the verifier generates a random bit string to use as the challenge. The verifier starts timing and transmits the challenge C via an RF channel. The prover simply replies via a sound channel with the same challenge C. The verifier accepts this if $L \in R$ and the elapsed time Δt is less than or equal to $d \cdot (c^{-1} + s^{-1}) + t_m$ where c and s are the speed of radio waves and sound, respectively, and d is the distance. This particular protocol is vulnerable when a proxy is placed close to the verifier, because echoing a nonce does not require any secret information from the real prover. The authors therefore proposed a variant for the case where the prover and verifier share a key. In the modified protocol the prover responds with $F_k(C)$ where F_k is a pseudo-random function.

This modified protocol still relies on an ultrasound channel that is susceptible to relay attacks. The prover also needs to calculate $F_k(C)$ during the exchange phase, which introduces inaccuracy and unreliability. Even though the prover transmits his expected processing time during setup this does not provide any extra security. A fraudulent prover can lie about the expected t_d to commit distance fraud or alternatively an attacker can advertise an expected processing delay that is sufficient to cover for a relay attack. This protocol does not allow for communication errors during the exchange stage.

Proving proximity to a trusted third party (2003)

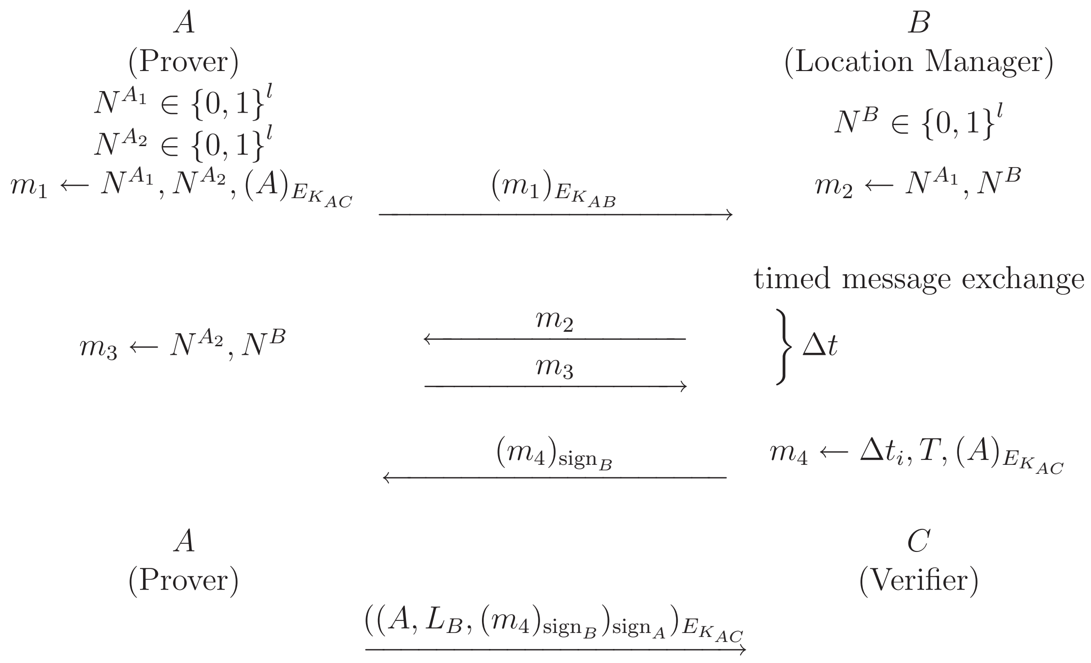

A
(Prover)
$N^{A_1} \in \{0,1\}^l$
$N^{A_2} \in \{0,1\}^l$
$m_1 \leftarrow N^{A_1}, N^{A_2}, (A)_{E_{K_{AC}}}$
$(m_1)_{E_{K_{AB}}}$
B
(Location Manager)
$N^B \in \{0,1\}^l$
$m_2 \leftarrow N^{A_1}, N^B$
timed message exchange
$m_3 \leftarrow N^{A_2}, N^B$
m_2
m_3
Δt
$(m_4)_{\text{sign}_B}$
$m_4 \leftarrow \Delta t_i, T, (A)_{E_{K_{AC}}}$
A
(Prover)
C
(Verifier)
$((A, L_B, (m_4)_{\text{sign}_B})_{\text{sign}_A})_{E_{K_{AC}}}$

Figure 5.5: Proximity proving to a trusted third party

Walters and Felten [165] proposed the distance-bounding protocol shown in Figure 5.5. This protocol is conceptually different, as the verifier C is not trying to bound the distance to the prover A. Instead, C wishes to confirm that A was within a certain distance from a location manager B at time T. In this case the two party distance-bounding protocol is executed by A and B, so strictly speaking B should be the 'verifier' but I prefer to keep the device descriptions as originally defined by the authors.

During the setup stage the prover sends message m_1, encrypted with shared key K_{AB}, to the location manager B. This message contains two nonces N^{A_1} and N^{A_2} generated by the prover, and also the prover's ID encrypted with a key, K_{AC}, shared with the verifier C. The significance of the nonces is as follows: The *start* nonce N^{A_1} is used by the B to indicate the start of the exchange stage and the *reply* nonce N^{A_2} is the response that will be sent by A. The location manager B also generates an *echo* nonce N^B. B now starts timing and transmits the *start* (N^{A_1}) and *echo* (N^B) nonces to which the prover answers with the *reply* (N^{A_2}) and *echo* (N^B) nonces. Once the exchange stage is completed the location manager issues the prover with a signed proximity certificate containing the round-trip time Δt, the current time T and the prover's ID, encrypted with K_{AC}, received during the setup stage. Finally, the prover constructs a message containing its ID, the proximity certificate and its location claim L_B ("I was near B"). The prover A then signs the message and encrypts it with a key shared with C before sending it to C.

Since all challenge and response data are decided upon during setup there is no need for processing during the exchange stage. The protocol prevents distance fraud as the prover needs to wait for the *echo* nonce before it can send a reply. A fraudulent prover that attempts to preemptively guess the echo nonce will succeed with probability 2^{-l}. The *start* nonce prevents an attacker from sending its own 'start nonce' earlier and recovering the reply before the location manager starts the exchange phase. A third party that simply attempts to preemptively guess the reply nonce will succeed with probability 2^{-l}. This protocol does not prevent terrorist attacks since the prover can give the replay nonce to a collaborating attacker without a penalty. This protocol does not allow for communication errors during the exchange stage.

Tang and Wu (2007)

The protocol proposed by Tang and Wu [158] incorporates a trusted third party and is based on an Elliptic Curve Zero-Knowledge Proof (ECZKP) and an Elliptic Curve Oblivious Transfer (ECOT). The server provides the starting time reference t_1 and issues the challenge to the prover, but the prover sends its response to the verifier who records the time of arrival t_2. It is assumed that the verifier and server have synchronised clocks, so the round trip time Δt is then calculated as $t_2 - t_1$. The authors provide extensive explanations, and security proofs, of the protocol and cryptographic primitives used. Due to space constraints, I will only explain the basic functions of the protocol using a simplified representation, as shown in Figure 5.6, and the notation as defined by the authors. The reader is therefore encouraged to consult the original publication for further information.

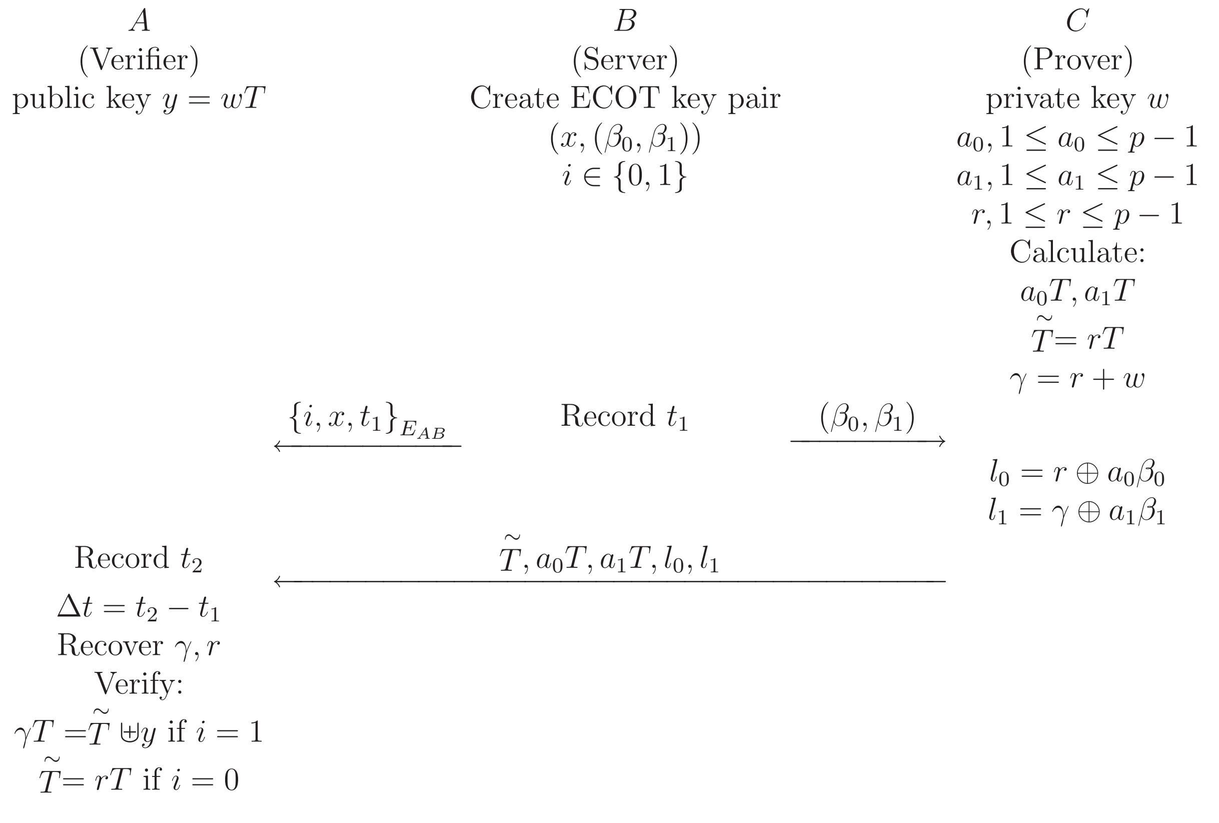


Figure 5.6: Distance-bounding scheme with a synchronised time server

It is assumed that the verifier has the prover's public key $y = wT$, where T is a point on an elliptic curve E over a Galois field F. During the setup stage the server creates an ECOT key pair $(x, (\beta_0, \beta_1))$ and a value i. For the ECZKP operation the prover generates a random number r and calculates $\widetilde{T}= rT$ and $\gamma = r + w$. For the ECOT operation the prover generates two random numbers a_0 and a_1 and calculates a_0T and a_1T. a_0, a_1 and r are values between 1 and $p - 1$, where p is the prime order of T. The ECOT operations is used to send γ and r to the verifier, while the ECZKP allows the prover to show that he knows w without revealing information about w to the verifier or an eavesdropper. At time t_1 the server transmits the ECOT public keys (β_0, β_1) to the prover. The server then sends the value of i, the ECOT key x and time t_1 via a protected channel to the verifier. The prover calculates the remaining ECOT strings l_0 and l_1 and then transmits to the verifier a packet containing $\widetilde{T}$, a_0T, a_1T, l_0 and l_1. The verifier records the time t_2 when this packet arrives and uses the measurement to determine Δt. The verifier completes the ECOT operation to retrieve γ and r and then verifies that the prover does indeed possess w using the ECZKP. $\uplus$ is defined as the point addition in E/F, which is the additive group derived from E and F.

The authors argue that this protocol is good for scalable systems since more than one prover can be bounded at the same time, i.e. the server sends the same ECOT public keys to multiple provers. The protocol prevents distance fraud as the prover has to wait for the ECOT keys before calculating and sending its response. The protocol does not prevent a "terrorist attack" as the prover could provide a proxy with a_0, a_1, r and γ before the exchange stage begins, which would allow the proxy to calculate l_0 and l_1. There is also some uncertainty introduced if the practical implementation is taken into consideration. During the exchange stage the prover needs to perform multiple scalar multiplication and XOR operations and the exchanged messages are very long. This introduces latency that could be exploited by an attacker and small variations in the processing time t_d, estimated by the authors to be in the millisecond range, which could cause errors in the distance estimate. The authors proposed that the protocol is run multiple times to obtain a reliable distance estimate. This protocol does not allow for communication errors during the exchange stage apart from the authors' suggestion that the verifier can tolerate high bit errors if it implements an automatic repeat request protocol that allows for repetition of the entire protocol.

5.4.2 Pre-commitment protocols

The protocols discussed in Section 5.4.1 perform multiple computations during the exchange stage. The possible variations in the processing delay t_d affects the time measurement, which in turn causes the distance bound to be unreliable. The processing delay should therefore be reduced to limit the inaccuracy in the time measurements. Some protocols suggest that the prover calculates its possible responses before the exchange stage. This is usually accomplished by using *pre-commitment* or *pre-computation*. Now the prover only has to choose a response R based on the challenge C received from the verifier during the exchange stage, so t_d is significantly reduced. These protocols generally propose that the function $C \rightarrow R$ is implemented using a simple XOR or table look-up operations to minimise variation in t_d. In protocols using pre-commitment the prover prepares possible responses during the setup stage. For example, the verifier generates a random challenge bit string, $C = (C_1, C_2, \ldots, C_l)$, while the prover generates a response string, $M = (M_1, M_2, \ldots, M_l)$. The prover commits to M, e.g. by transmitting a collision-resistant message authentication code $h(K, M)$. The verifier then sends one C_i after another, which the prover receives as C'_i. It then instantly replies with a bit $R_i = C'_i \oplus M_i$, which is calculated by XOR-ing each received challenge bit with the corresponding bit of M. Finally the prover reveals M and authenticates C', i.e. prover sends $MAC(C')_K$ to verifier.

Making the prover echo the challenge C_i will probably result in the lowest processing delay. This is unfortunately not a secure options as it would not prevent a relay attack. The attacker could relay communication during the setup stage and then during the exchange stage the proxy-prover simply echoes the challenges back to the verifier. Since the proxy-prover is in close proximity to the verifier the measured RTT translates to a distance estimate that is within the acceptable upper bound. The proxy-prover also transmits the challenges to the proxy-verifier, which sends them to the real prover. During the verification stage the attacker once again relays the communication from the prover so the verifier receives a valid $MAC(C')_K$. As a result the verifier believes that the real prover is present within the allowable distance. It is therefore important that the prover is required to generate a response. If the prover, however, only generated a response string R during setup and was expected to transmit R_i after receiving C_i he could preemptively transmit R_i before receiving the challenge. The response string M is therefore introduced to prevent distance fraud. Making the prover calculate R_i based on C_i and M_i forces him to wait until he has received the challenge before responding. In addition the commitment on M prevents the prover from sending a random bit R_i early and then setting $M_i = C'_i \oplus R_i$ after receiving C'_i. Authenticating C' prevents an attacker from preemptively sending a fake challenge bit to the prover in an attempt to discover M_i before the verifier issues the real challenge. Alternatively a prover can generate two response strings, $M^1 = (M_1^1, M_2^1, \dots, M_l^1)$ and $M^2 = (M_1^2, M_2^2, \dots, M_l^2)$, commit to both and then use a 1-bit lookup table to reply with bit $R_i = M_i^{C'_i}$. Protocols using pre-computation are discussed in Section 5.4.3.

Brands-Chaum (1993)

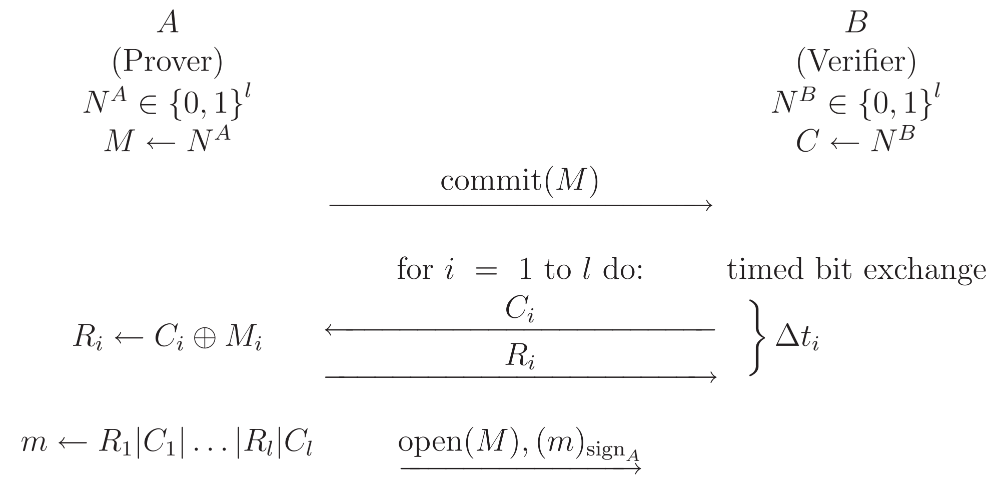


Figure 5.7: Brands-Chaum distance-bounding protocol

Brands and Chaum [13] described the first distance-bounding protocol based on timing the single-bit round-trip time in a cryptographic challenge-response exchange as shown in Figure 5.7. 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 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.

This protocol protects against distance fraud and relay attacks. A fraudulent prover or a third party attacker that attempts to preemptively guess all the response bits R_i will succeed with probability 2^{-l}. Similarly a third party, who prematurely requests R_i from the prover, by supplying it with guessed challenges C'_i, will succeed with probability 2^{-l} without being detected. The protocol fails if a single bit error occurs during the exchange stage, since the verification signature will be incorrect and not accepted by the verifier. The protocol does not protect against terrorist attacks because the prover suffers no penalty for releasing M to a collaborating attacker.

Mutually authenticated distance bounding (2003)

Čapkun, Buttyán and Hubaux proposed a distance-bounding protocol (MAD) [20], which modifies the Brands-Chaum protocol to allow two devices participating to bound the distance to the other party simultaneously. The protocol is shown in Figure 5.8.

In the setup stage both parties generate a random bit string of length l. This bit string is used to generate both the response and the challenge. Both parties commit to their respective response string before the exchange phase starts. One device, A, then transmits the first bit of his response string as a challenge bit $\alpha_1 = M_1^A$, to which the other device, B, responds immediately with $\beta_1 = \alpha_1 \oplus M_1^B$. Device A then immediately responds with $\alpha_2 = M_2^A \oplus \beta_1$. Device A measures the time taken to exchange α_i and β_i while device B measures the time taken to exchange β_i and $\alpha_{(i+1)}$. During verification both devices reveal their respective response strings M^A and M^B. Each device also transmits the message authentication code of a message containing the ID of A, the ID of B, its own response string and what it believes the other device's response string to be (calculated from XORing the sent challenges and received responses). As with the Brands-Chaum protocol, MAD protects against distance fraud and relay attacks but not terrorist attacks. A single bit error causes the protocol to fail.

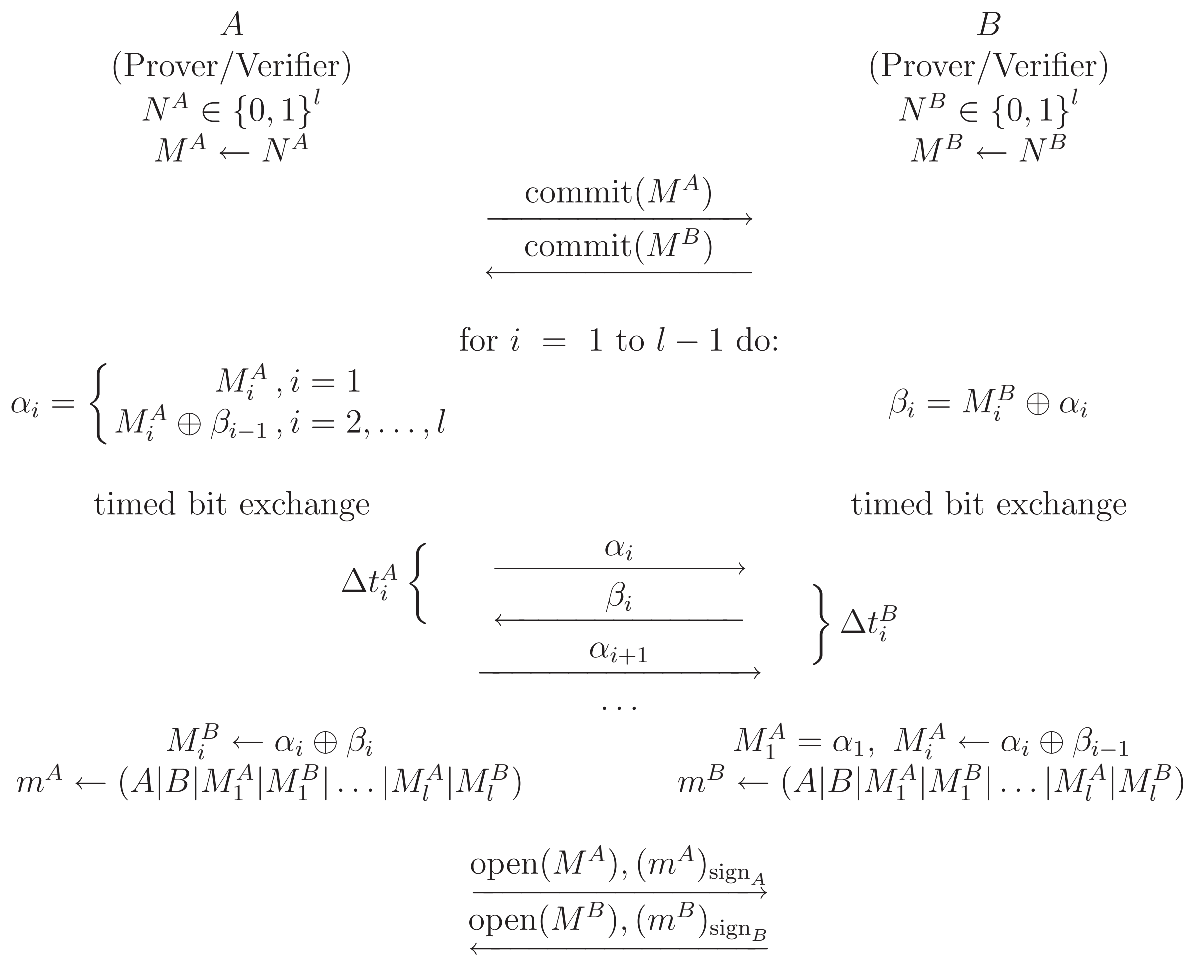

A
(Prover/Verifier)
$N^A \in \{0,1\}^l$
$M^A \leftarrow N^A$
B
(Prover/Verifier)
$N^B \in \{0,1\}^l$
$M^B \leftarrow N^B$
$\xrightarrow{\text{commit}(M^A)}$
$\xleftarrow{\text{commit}(M^B)}$
for $i = 1$ to $l-1$ do:
$\alpha_i = \begin{cases} M_i^A, i = 1 \\ M_i^A \oplus \beta_{i-1}, i = 2, \ldots, l \end{cases}$
$\beta_i = M_i^B \oplus \alpha_i$
timed bit exchange
timed bit exchange
Δt_i^A
$\xrightarrow{\alpha_i}$
$\xleftarrow{\beta_i}$
$\xrightarrow{\alpha_{i+1}}$
Δt_i^B
$\ldots$
$M_i^B \leftarrow \alpha_i \oplus \beta_i$
$m^A \leftarrow (A|B|M_1^A|M_1^B|\ldots|M_l^A|M_l^B)$
$M_1^A = \alpha_1,\ M_i^A \leftarrow \alpha_i \oplus \beta_{i-1}$
$m^B \leftarrow (A|B|M_1^A|M_1^B|\ldots|M_l^A|M_l^B)$
$\xrightarrow{\text{open}(M^A), (m^A)_{\text{sign}_A}}$
$\xleftarrow{\text{open}(M^B), (m^B)_{\text{sign}_B}}$

Figure 5.8: Mutually authenticated distance-bounding protocol (MAD)

Čapkun and Hubaux (2005)

Čapkun and Hubaux proposed a distance-bounding protocol for use in secure positioning [22]. They modify the Brands-Chaum protocol by changing the exchange stage. Instead of multiple single-bit exchanges there is now only a single message exchange involving a multi-bit challenge and response. The protocol is shown in Figure 5.9.

In the setup stage the prover commits to a response string M and the verifier generates a challenge C. The verifier then sends C to which the prover replies with $R = C \oplus M$. The prover transmits R with the least significant bit (LSB) first and C is transmitted with the most significant bit (MSB) first. During verification the prover reveals M and transmits a signature of his ID, R and C while the verifier checks that $M = C \oplus R$. By making the first bit of R dependent on the last bit of C the protocol ensures that the prover must wait until it received all of C before replying. Like the other protocols, this one fails when a bit error occurs during the exchange phase. Even with the similarities to the Brands-Chaum protocol, the implementation of the exchange stage leaves the protocol vulnerable to a guessing attack, which a fraudulent prover or third party can use to commit distance fraud or relay attacks. I describe the guessing attack in detail in Section 5.4.4.

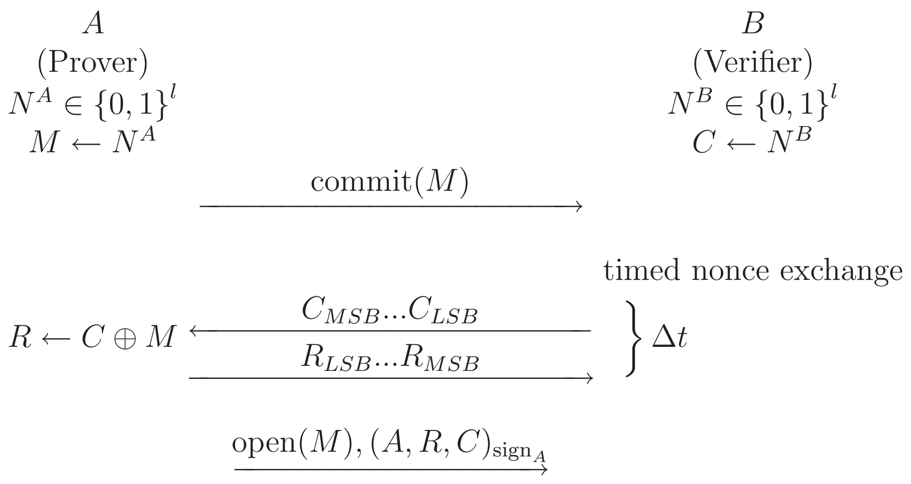


Figure 5.9: Čapkun-Hubaux distance-bounding protocol

Distance-bounding proof of knowledge (2005)

Bussard and Bagga proposed a protocol that would discourage the prover from collaborating in a "terrorist attack" [18]. Their basic assumption is that the prover has something highly valuable, in this case his private key x, that he would not want to give to anyone. The protocol is then formulated in such a way that the prover will reveal his valuable secret if he discloses both response strings to an attacker. The protocol is shown in Figure 5.10. The reader should note that, due to space constraints, I will only discuss the basic functioning of the protocol and reasoning behind its design. For additional information about the properties of the cryptographic primitives *commit*, v, Γ and Ω in addition to details about the zero-knowledge proof of knowledge please consult [17, pp 93–105].

It is assumed that the prover has a private key x and a public key $y = \Gamma(x)$. During the setup stage the prover generates a random bit string N^{A_1} of length l, which is used as the first response string M^1. The prover also generates a second response string M^2 by encrypting the private key x with the first response string M^1 using a publicly known symmetric encryption method. The prover then commits to both M^1 and M^2 before the exchange stage using bitwise commitments $m_{(M^1,i)}$ and $m_{(M^2,i)}$ for $i = 1$ to l. The *commit* function used cannot simply bind the prover to a response string and then disclose the string during the verification stage. If this was the case *open*(M^1) and *open*(M^1) would disclose the complete M^1 and M^2 strings, which will allow an attacker or verifier to recover x. The bitwise commitment allows the prover to only open commitments on the bits from M^1 and M^2 that were used during the exchange stage. The authors have also designed the *commit* in such a way that it is tied to the prover secret key. This allows the verifier to confirm during the verification stage that the prover generated the commitments.

Before the exchange stage the verifier generates a random bit string N^B, of length l, to use as the challenge string C. The verifier then transmits one challenge bit C_i at a time (for all $i = 1, \ldots, l$), to which the prover responds immediately with R_i. The verifier times the round-trip delay between sending each bit C_i and receiving the corresponding response bit R_i. This protocol uses a 1-bit lookup table instead of an XOR operation to determine the response bits R_i. During each bit exchange the prover replies with M_i^1 if $C_i = 0$ and M_i^2 if $C_i = 1$. This way of determining the response, on average, only reveals $\frac{l}{2}$ bits of each response string during protocol execution. It is therefore not possible for an attacker, or the verifier, to obtain any significant information about the private key unless the prover chooses to disclose both M^1 and M^2. During verification the prover selectively opens commitments on bits from M^1 and M^2 that were used to determine responses during the exchange stage and the verifier confirms that the commitments are valid. Finally the prover proves to the verifier that he generated the commitments, that the generated commitments correspond to a unique private key and that this private key corresponds to the his public key y, which is used by the verifier to authenticate the prover. This protocol protects against distance fraud and relay attacks, while also forcing the prover to reveal a valuable secret if it participates in a "terrorist attack" and discloses M^1 and M^2. If the prover collaborates the third party would simply be able to decrypt M^2 with M^1 and recover the prover's private key x.

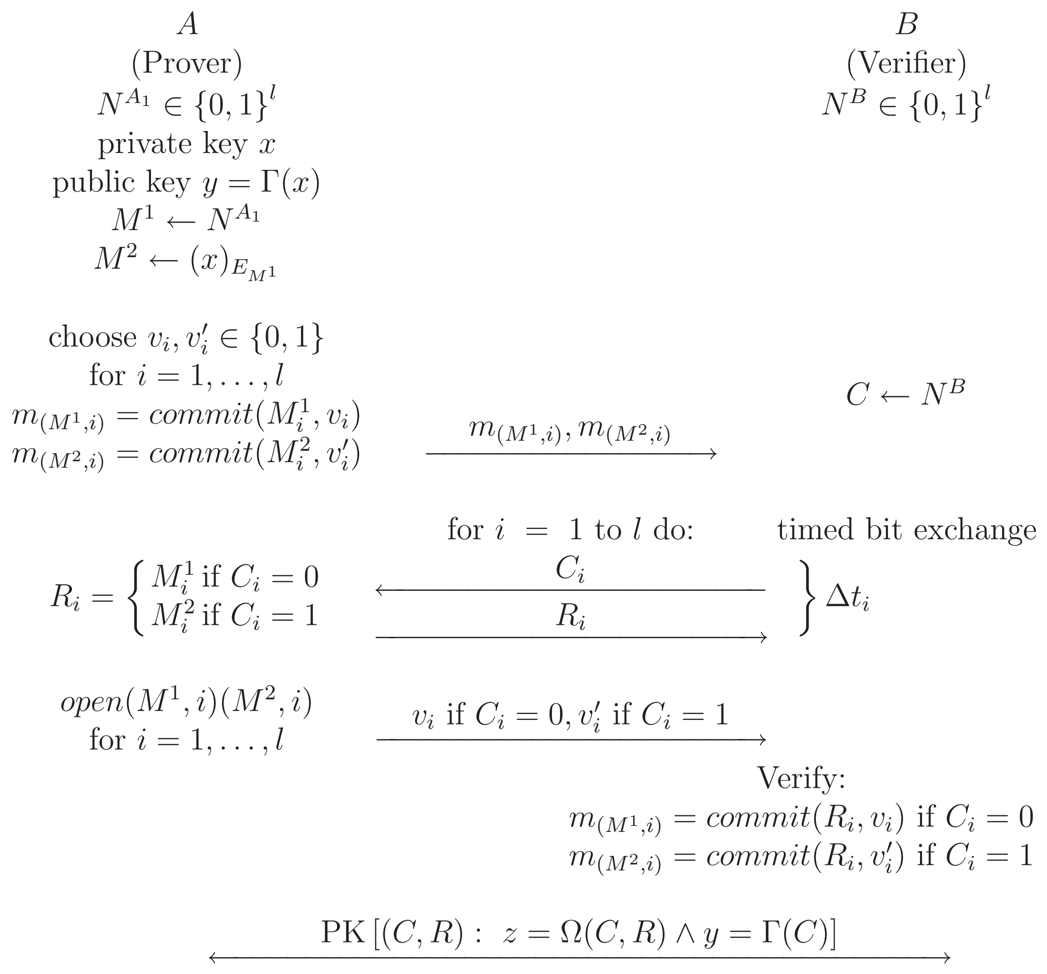

A
(Prover)
$N^{A_1} \in \{0,1\}^l$
private key x
public key $y = \Gamma(x)$
$M^1 \leftarrow N^{A_1}$
$M^2 \leftarrow (x)_{E_{M^1}}$
B
(Verifier)
$N^B \in \{0,1\}^l$
choose $v_i, v_i' \in \{0,1\}$
for $i = 1, \ldots, l$
$m_{(M^1,i)} = commit(M_i^1, v_i)$
$m_{(M^2,i)} = commit(M_i^2, v_i')$
$m_{(M^1,i)}, m_{(M^2,i)}$
$C \leftarrow N^B$
for $i = 1$ to l do:
timed bit exchange
$R_i = \begin{cases} M_i^1 \text{ if } C_i = 0 \\ M_i^2 \text{ if } C_i = 1 \end{cases}$
C_i
R_i
Δt_i
$open(M^1, i)(M^2, i)$
for $i = 1, \ldots, l$
v_i if $C_i = 0$, v_i' if $C_i = 1$
Verify:
$m_{(M^1,i)} = commit(R_i, v_i)$ if $C_i = 0$
$m_{(M^2,i)} = commit(R_i, v_i')$ if $C_i = 1$
$\mathrm{PK}\left[(C,R):\ z = \Omega(C,R) \wedge y = \Gamma(C)\right]$

Figure 5.10: Distance-bounding proof of knowledge protocol

5.4.3 Pre-computation protocols

In protocols using pre-computation the prover and the verifier calculates the possible response strings before the exchange stage starts. For example, the verifier and the prover first exchange nonces N_V and N_P. Both the prover and the verifier then use a pseudo-random function F and a shared key K in order to calculate two n-bit response strings M^1 and M^2:

$$(M_1^1, M_2^1, M_3^1, \ldots, M_l^1, M_1^2, M_2^2, M_3^2, \ldots, M_l^2) \ := \ F_K(N_V, N_P)$$

Since the verifier also knows M^1 and M^2 at this stage the prover is effectively committed to two response strings without explicitly making a commitment during setup. As a result the prover does not have to open his commitment during the verification stage, thereby decreasing the data that needs to be transmitted. The verifier also generates a random challenge bit string $C_1, C_2, \ldots, C_l$ and starts the exchange stage. The prover's reply bit $R_i = M_i^{C_i'}$ to each C_i' received from the verifier is the result of a 1-bit table lookup in M^1 or M^2, selected by the received challenge bit C_i' (for $1 \leq i \leq l$). The verifier checks whether the R_i' bits that it received matches the locally calculated expected values. Similarly the prover and verifier can calculate a single response string M^1 and use an XOR operation to determine the response R.

Hancke and Kuhn (2005)

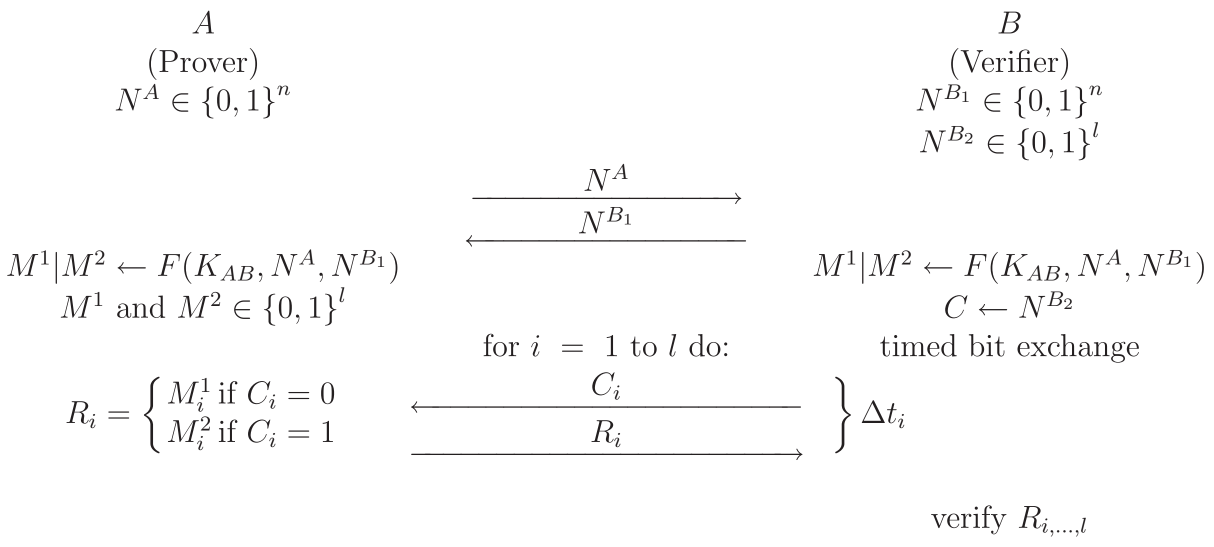


Figure 5.11: Hancke-Kuhn distance-bounding protocol

Kuhn and I proposed a protocol that minimizes data transmission and is resistant to bit errors during the exchange stage [64]. The protocol is shown is Figure 5.11. Our protocol uses a 1-bit lookup table, a concept proposed independently by us and Bussard and Bagga (see previous section), and pre-computation instead of commitment. We assume that the exchange stage is performed using a 'fast' channel and that other communication is sent over a 'slow' error-corrected channel. The 'fast' channel provides adequate timing resolution for distance bounding and uses modulation techniques resistant to relay attacks, but at the same time it is susceptible to bit errors. This protocol is discussed in more detail later in Section 5.5.

During the setup stage the verifier and prover exchange nonces, which they use to calculate the response strings. The verifier generates a random nonce N^{B_1} of length n in addition to a random bit string N^{B_2} of length l. The nonce is then transmitted to the prover and the bit string kept as the challenge. The prover also generates a nonce N^A, which is transmitted to the verifier. The prover and verifier both calculate $F(K_{AB}, N^{B_1}, N^A)$, where F is a pseudo-random function, and split the result into two response strings M^1 and M^2, both of length l. Most of the processing it therefore done before the exchange stage starts. Since the verifier can also calculate the response strings during setup the prover is effectively bound to using these response strings without explicit commitment. This allows for less data transmission since the prover does not need to open his commitment and reveal his response strings during the verification stage. During the exchange stage the verifier transmits one challenge bit C_i at a time (for all $i = 1, \dots, l$), to which the prover responds immediately with $R_i = M_i^1$ if $C_i = 0$ and $R_i = M_i^2$ if $C_i = 1$. The verifier times the round-trip delay between sending each bit C_i and receiving the corresponding response bit R_i. There is no data transmitted during the verification stage. As the verifier knows M^1, M^2 and C it can calculate the expected response bits R_i. The verifier then only checks whether at least k of the l response bits that it received match the expected response bits it calculated earlier. By specifying threshold k the protocol still works even if $l - k$ bit errors occur. The values k and l are security parameters and should be chosen to ensure acceptable false acceptance and false rejection probabilities. F should be chosen such that it can generate an output of length $2l$.

Our protocol specifies that both the prover and verifier generate a nonce. The main reason for this is that the protocol is designed to accommodate a resource-limited prover that would not be able to generate a strong random number to use as a nonce and which is expected to function using an untrusted clock. If $F(K_{AB}, N^A)$ was used to generate M^1 and M^2 an attacker would be able to keep running the protocol with the prover using a challenge string of all zeros, thus reading out M^1, until the nonce repeated. The attacker then issues a challenge string of all ones, thereby reading out M^2. The attacker now knows both M^1 and M^2 for a specific N^A so he can transmit this nonce to any number of verifiers and pass the distance bound by providing all the correct responses. If the prover does not have the ability to generate any nonce the response strings can be determined by calculating only $F(K_{AB}, N^{B_1})$. This variation of the protocol should, however, only be used if it is not possible for a proxy-verifier to run the protocol with the prover twice, using the same nonce N^{B_1} each time and thereby recovering both M^1 and M^2, before the verifier starts its exchange stage. For a more detailed explanation of this attack please refer to Section 5.5.2. This protocol protects against distance fraud, as the prover has to wait for the challenge before responding, and relay attacks. The main weakness is that an attacker can send his own challenge before the verifier and recover half of the response strings' content. This means that the attacker can guess the correct reply R_i with probability of $\frac{3}{4}$. For a more detailed explanation of this attack please refer to Section 5.5.2.

Void-challenge protocol (2006)

Munilla, Ortiz and Peinado proposed a protocol, similar to Hancke-Kuhn, that aims to provide additional security during the exchange stage by randomising the time at which the verifier sends the challenge bit [115]. The protocol is shown in Figure 5.12. The exchange stage is split into l time slots and the prover knows in which slot the verifier will issue a challenge. This discourages an attacker from gathering response string data, by sending a challenge before the verifier, since the prover could detect the attack if it receives an unexpected challenge. This means that the attacker can only guess the correct reply R_i with a probability of $\frac{1}{2}$. The prover does not know the value of the challenge, only the time slot, so it cannot commit distance fraud by preemptively sending a response.

The verifier generates a random nonce N^{B_1} of length n and a random bit string N^{B_2} of length l. The nonce is then transmitted to the prover and the bit string kept as the challenge. The prover generates a random nonce N^A of length n and transmits it to

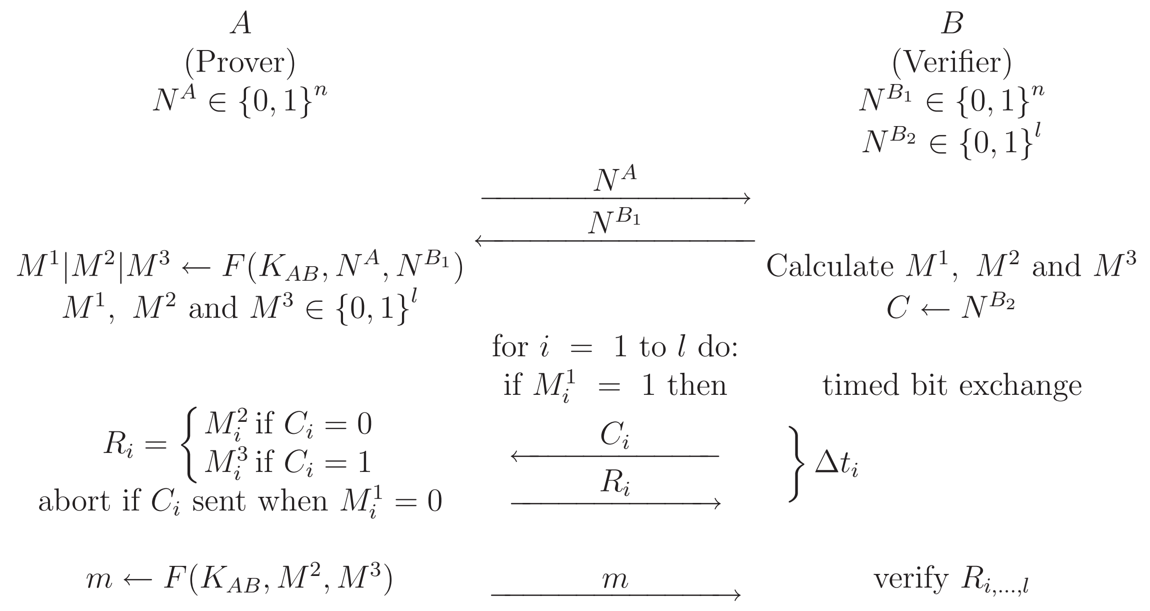

A
(Prover)
$N^A \in \{0,1\}^n$
B
(Verifier)
$N^{B_1} \in \{0,1\}^n$
$N^{B_2} \in \{0,1\}^l$
N^A
N^{B_1}
$M^1|M^2|M^3 \leftarrow F(K_{AB}, N^A, N^{B_1})$
$M^1,\ M^2$ and $M^3 \in \{0,1\}^l$
Calculate $M^1,\ M^2$ and M^3
$C \leftarrow N^{B_2}$
for $i = 1$ to l do:
if $M_i^1 = 1$ then
timed bit exchange
$R_i = \begin{cases} M_i^2 \text{ if } C_i = 0 \\ M_i^3 \text{ if } C_i = 1 \end{cases}$
abort if C_i sent when $M_i^1 = 0$
C_i
R_i
Δt_i
$m \leftarrow F(K_{AB}, M^2, M^3)$
m
verify $R_{i,\ldots,l}$

Figure 5.12: Void-challenge distance-bounding protocol

the verifier. Both parties then calculate $F(K_{AB}, N^{B_1}, N^A)$, where F is a pseudo-random function, to determine three bit strings M^1, M^2, M^3, each with length l. M_1 indicates the time slots in which the challenges will be sent while M_2 and M_3 are the response strings. The exchange stage is broken up into l time-slots and the verifier will transmit a challenge bit C_i only during time slot numbered i if $M_i^1 = 1$. The prover responds immediately with $R_i = M_i^2$ if $C_i = 0$ and $R_i = M_i^3$ if $C_i = 1$. The verifier times the round-trip delay between sending each bit C_i and receiving the corresponding response bit R_i. During verification the prover confirms the values it calculated for M^2 and M^3. The pseudo-random function should be chosen such that it generates an output of length $3l$.

For this protocol to work the communication channel would need three symbols, which would allow the prover to distinguish between challenge ‘0’, challenge ‘1’ and no challenge. Only aborting if the prover receives $C_i = 1$ when $M_i^1 = 0$, in other words when the symbol for no challenge is the same as for challenge ’0’, is not sufficient. In this case a prover will not be able to detect an attacker’s challenge if he sent a ‘0’ in a time slot where a challenge is not expected. This means that the attacker would be able to gather all of M^2 without being detected, by preemptively challenging the prover with only ‘0’ challenges during every time slot. This protocol protects against distance fraud and relay attacks. This protocol can be made resistant to communication errors in a similar way to the previous protocol by specifying acceptance thresholds in the prover and verifier.

Reid, Nieto, Tang and Senadji (2006)

Reid, Nieto, Tang and Senadji proposed a protocol that discourages “terrorist attacks” assuming both parties have the ability to perform symmetric encryption [137]. The protocol is shown in Figure 5.13. The basic principle, that the prover has a valuable secret that he does not want to reveal, is similar to the Bussard-Bagga protocol, but in this case the protocol requires no public-key cryptography. Like the Hancke-Kuhn protocol it uses pre-computation and no data is exchanged during the verification stage.

The verifier generates a random nonce N^{B_1} of length n and a random bit string N^{B_2}

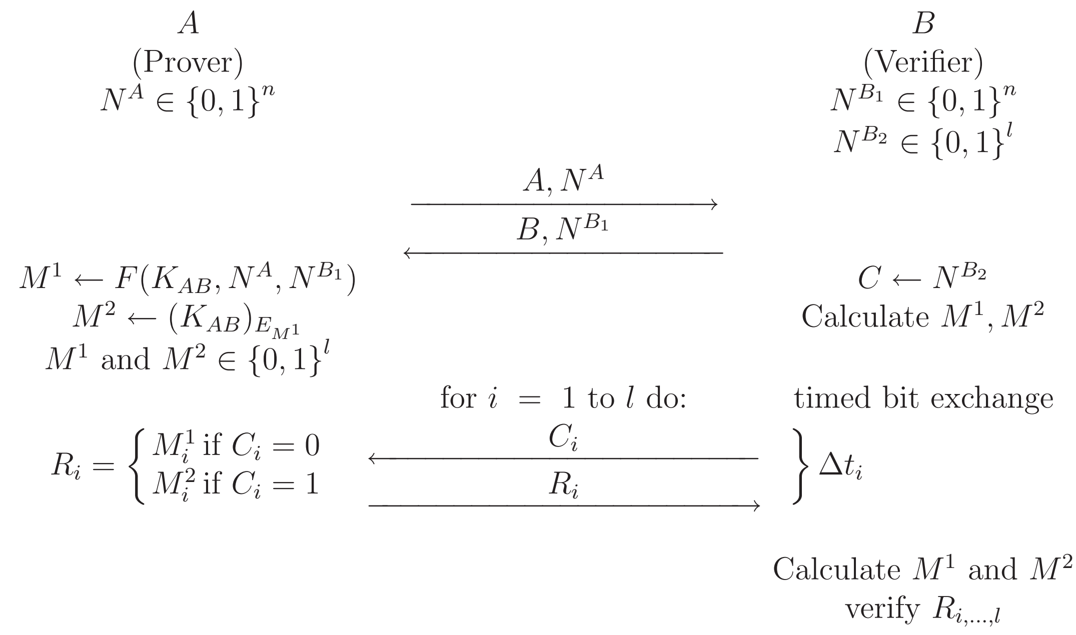


Figure 5.13: "Terrorist-resistant" distance-bounding protocol

of length l. The nonce is then transmitted to the prover and the bit string kept as the challenge. The prover generates a random nonce N^A of length n and transmits it to the verifier. Both parties then calculate $F(K_{AB}, N^{B_1}, N^A)$, where F is a pseudo-random function, to determine M^1 with length l. M^2 is determined by encrypting K_{AB} with M^1, which means that if a prover reveals the response strings he also reveals his valuable shared key K_{AB}. The verifier then transmits one challenge bit C_i at a time (for all $i = 1, \dots, l$), to which the prover responds immediately with $R_i = M_i^1$ if $C_i = 0$ and $R_i = M_i^2$ if $C_i = 1$. The verifier times the round-trip delay between sending each bit C_i and receiving the corresponding response bit R_i. During the verification stage the verifier checks whether the prover's responses matches the expected values it calculated locally. The protocol also allows an attacker to send his own challenge and recover half of the response strings' content before the verifier starts the exchange stage. This protocol protects against distance fraud and relay attacks while discouraging the prover to participate in a "terrorist attack". This means that the attacker can guess the correct reply R_i with a probability of $\frac{3}{4}$.

The authors state that their protocol can be made resistant to bit errors by using the same method used for the Hancke-Kuhn protocol. If this is the case, the verifier checks whether at least k of the l response bits that it received matches the expected R it calculated, so the protocol still works even if $l - k$ bit errors occur. The values k and l should be chosen to ensure acceptable false acceptance and false rejection probabilities. The verifier only checks whether at least k of the l response bits that it received matches the expected R it calculated. By specifying threshold k the protocol still works even if $l - k$ bit errors occur. If the verifier allows for bit errors the prover might be able to collaborate in a "terrorist attack" without revealing K_{AB}. The authors assume that if the prover hands over M^1 and M^2 the attacker can recover the prover's secret key K_{AB}. However, the prover could deliberately change bits in M^1 and M^2 while still allowing the attacker to pass the distance bound because the verifier expects some bit errors. The prover changes enough bits to make a brute force search on K_{AB} difficult without causing more than $l - k$ bit errors in the third-party attacker's response. For example, the prover knows that the verifier will allow 5 bit errors so he intentionally modifies a total of 5 bits in M^1 and M^2. As a result, some of the attacker's responses could be incorrect since he uses these response strings to calculate the responses. The exact number of incorrect responses depends on the challenges, and whether a modified bit is actually required to calculate the response, but the maximum number that can occur is 5. The verifier will accept the responses, as he is allowing for up to 5 bit errors, and evaluate that the distance bound holds. The prover has therefore successfully colluded with the attacker to circumvent the distance bound. The attacker, however, cannot recover the prover's key before he identifies all the modified bits in M^1 and M^2. In my example, the attacker will have a $1/\binom{2l}{5}$ chance of guessing the correct K_{AB} if he knows that there are 5 modified bits.

Meadows, Syverson and Chang (2006)

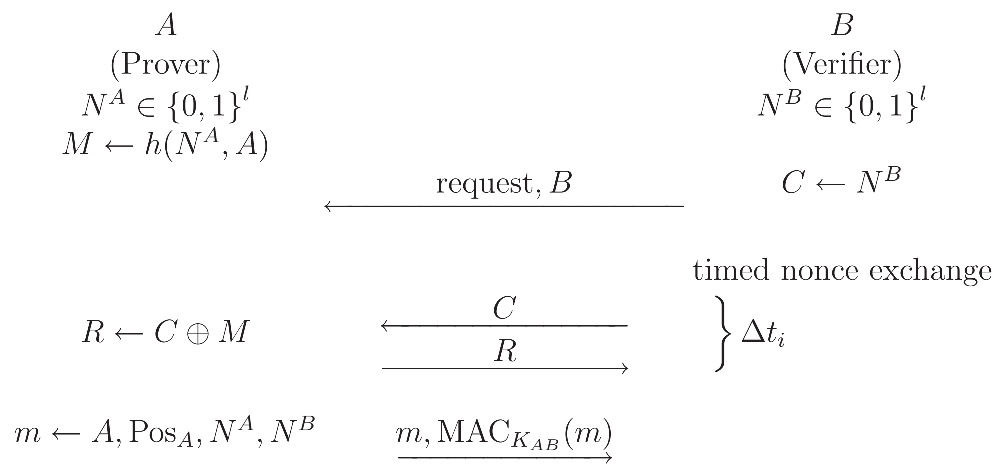


Figure 5.14: Authenticated distance-estimation protocol

Meadows, Syverson and Chang proposed a distance-bounding protocol that aims to reduce message complexity [109]. The protocol also does not require a shared key before the verification stage. The protocol was designed in this way to allow a sensor node to make initial estimates of the distances to other nodes, and determine its neighbors before keys are exchanged. The protocol starts with the verifier B sending out a request along with its identity. The verifier then transmits a challenge N^B and the prover answers with $F(N^A, N^B, A)$. The authors state that function F used to calculate the response should be chosen so that the verifier is able to verify that the response was created using N^A, N^B and A. They suggest $F(N^A, N^B, A) = N^B|N^A|A$, $F(N^A, N^B, A) = N^B|A \oplus N^A$ or $F(N^A, N^B, A) = N^B \oplus h(N^A, A)$ as three possible functions. One of these protocol variations is shown in Figure 5.14. In this case the prover pre-calculates the response string $M = h(N^A, A)$. The verifier then transmits a challenge N^B and the prover answers with $N^B \oplus M = N^B \oplus h(N^A, A)$. Once the verifier has completed the distance bound it will exchange a key K_{AB} with the prover, or use a key exchanged earlier, and commence with the verification stage. During this stage the prover transmits its identity, N^A, N^B and its position Pos_A.

In this protocol variation only the prover calculates the response string and there is no explicit commitment before the exchange stage. The prover is, however, prevented from preemptively sending a response R' and then verifying with a response string $R' \oplus N^B$, since it is not feasible to find a suitable value for N^A, such that $h(N^A, A)$ is equal to

$R' \oplus N^B$, before the verification stage. Distance fraud is also prevented for the remaining variations since the prover has to wait for N^B before sending the response. The use of multi-bit data packets during the exchange stages makes the protocol vulnerable to attacks at the physical layer of the communication medium and verification will also fail if bit errors occur.

5.4.4 Protocol weaknesses

The protocols discussed in Sections 5.4.1 to 5.4.3 have both advantages and disadvantages in terms of practical implementation, reliability, accuracy and security. Although each proposal should be seen in its own context it is noticeable that several protocols exhibit the same weaknesses in timing accuracy, latency and bit errors in the rapid exchange stage.

Timing precision and latency during the exchange stage

The exchange stage is the most critical since the timing information is used to calculate the physical distance bound. The accuracy of the distance bound is influenced by the precision of the timing mechanism, properties of the communication channel and the processing delay t_d between receiving a challenge and sending the response. Careful consideration must therefore be given to how the communication channel used for the exchange is implemented. The protocol and underlying communication must keep t_d to a minimum to ensure an accurate distance measurement. Any unnecessary delays in transmission of data must also be avoided since an attacker can exploit any latency to his advantage.

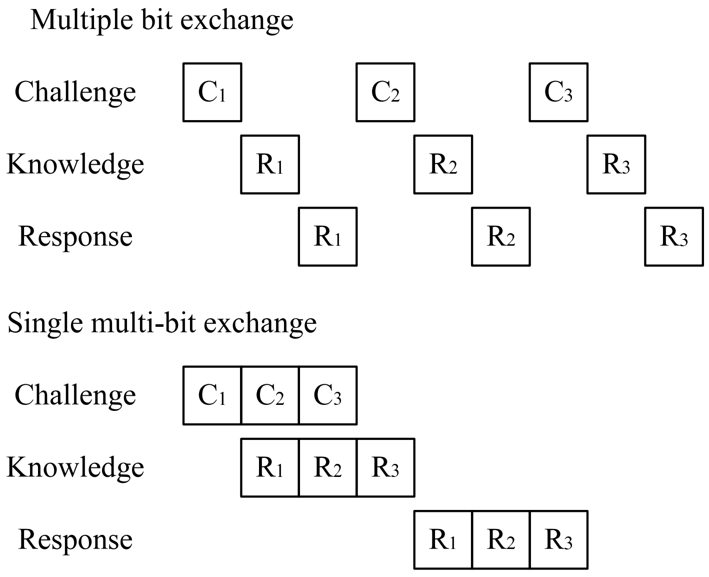


Figure 5.15: Comparison between a multiple-bit exchange and a single multi-bit exchange. 'Knowledge' indicates the time at which the prover knows the response bit R_i.

The timed authentication protocols in Section 5.4.1 perform lengthy computations, e.g. the generation of a signature or pseudo-random output, during the timed exchange stage. Not only does this introduce an inaccuracy into the distance calculation if t_d varies, but a malicious prover with high performance hardware can extract a time advantage by performing these operations faster. The proposals in Sections 5.4.2 and 5.4.3 perform all cryptographic operations before the exchange stage so these successfully minimize t_d. Some of these protocols, however, use a single multi-bit exchange instead of the multiple single-bit exchanges described originally in the Brands-Chaum protocol. Protocols that exchange multi-bit strings, or nonces, are vulnerable to attacks exploiting latency in the underlying communication channel. In contrast a multiple single-bit exchange provides the highest time resolution, as it depends only on propagation time, pulse width and processing delay of a single bit. Multiple timed bit exchanges may appear inefficient but multiple measurements increase accuracy and confidence. The difference between the two types of exchange is shown in Figure 5.15. In this example I assume that the response is determined at the bit level, i.e. C_i is calculated from R_i.

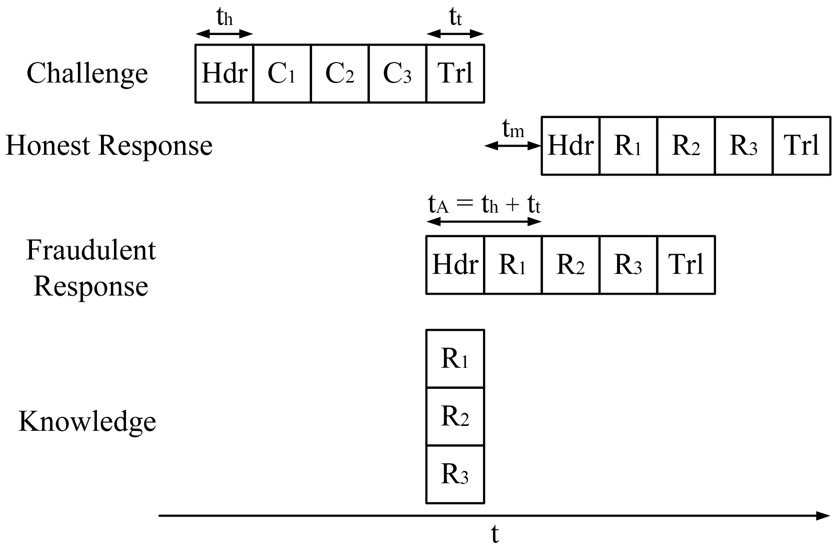


Figure 5.16: Extra data, used for formatting and redundancy, introduces latency that the attacker can exploit.

Communication channels usually transmit multi-bit data packets rather than single bits, so the nonce exchange method is easier to implement with off-the-shelf components. The problem is that these channels usually add some extra formatting information, such as trailers, headers and error-correction measures, that introduce latency. This could be as simple as adding a parity, start and stop bit to the sent data. An attacker may not be restricted by regular implementations and could reduce latency introduced by the communication layers to commit distance fraud. In the example shown in Figure 5.16 a verifier takes a round-trip time measurement t_m from the time it finished transmitting the challenge until it receives the first bit of the response. An attacker can gain time t_A equal to $t_\mathrm{h} + t_\mathrm{t}$ if the verifier expects the prover to strictly adhere to the communication protocol. The attacker ignores the data trailer and starts calculating its response while preemptively transmitting the header of the return data. This example only shows the effect of adding extra formatting, therefore I assume that the attacker has to wait for the last challenge bit before calculating the response.

Even if a nonce was exchanged without any formatting, or error correction, an attacker could still gain a timing advantage. For the purpose of illustrating this attack I assume that a single multi-bit exchange takes place and that the response is determined at the bit level, i.e. C_i is calculated from R_i. For example, this is the case in the protocol proposed by Meadows, et al. in Figure 5.14. The verifier takes a round-trip time measurement t_m

from the time it finished transmitting the challenge until it receives the first bit of the response. If the prover is honest he will adhere to the rules of the protocol and wait until he receives the entire challenge before calculating and sending his response. However, it is possible to calculate response bit R_i as soon as bit C_i is received. As a result, the prover knows some of the correct response bits before receiving all of the challenge. As shown in Figure 5.17, a fraudulent prover can actually start to transmit correct response bits at time t_r. At this time the prover has received C_1 and calculated R_1, which means he is ready to start his response. By the time the prover has transmitted R_1 he has already received C_2 and calculated R_2 so he is ready to send the next response bit. The prover continues calculating the next response bit early as the rest of the challenge bits are received. As a result, the fraudulent prover manages to preemptively answer with a correct set of response bits T_A earlier than expected. As the prover gains T_A he can effectively obscure any additional propagation time resulting from him being further away, thereby circumventing the verifier's distance bound. In this case, the fraudulent prover can be up to $c \cdot (T_\mathrm{A}/2)$ further away from the verifier and still appear within the allowable distance.

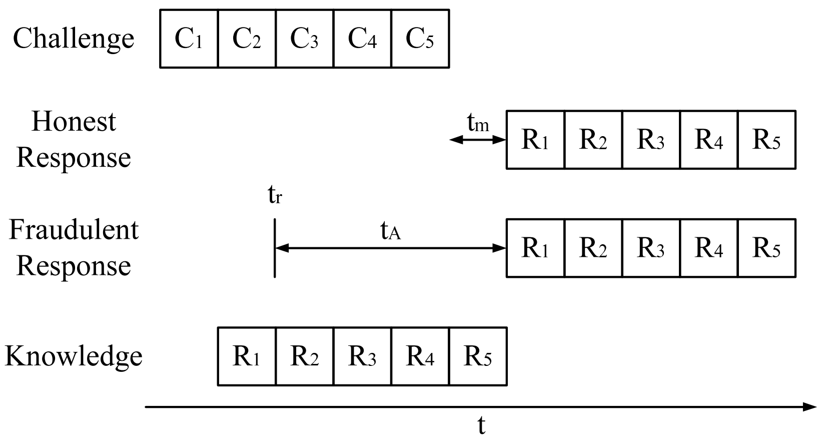


Figure 5.17: Timing attack on nonce exchange where the prover can respond at any time after t_r once it calculated R_1. The exchange timing is shown from the perspective of the verifier, i.e. the position of each block indicates the relative time at which it is observed by the verifier.

The protocol by Čapkun and Hubaux, as shown in Figure 5.9, tries to fix this problem by forcing the prover to wait until the last bit of the challenge before sending a response. It does this by making the first bit of the response dependent olast bit of the challenge. The challenge is transmitted most significant bit first while the response is transmitted least significant bit first. The prover therefore has to wait for the least significant bit C_l to arrive before it can respond with the first response bit. This protocol is vulnerable to a guessing attack where an attacker can guess the value for the last bit transmitted by the verifier and preemptively transmit a response. The attacker guesses correctly with probability $\frac{1}{2}$ and gains a timing advantage of up to twice the bit period, as shown in Figure 5.18. The advantage gained depends on the bit period t_B for the channel. An exchange of l single-bit challenges reduces an attacker's chances of guessing the correct response to 2^{-l}. A single l-bit message can be attacked successfully with a probability of only $\frac{1}{2}$. An attacker can guess more than one response and shorten t_m by $2t_\mathrm{B} \cdot g$ with probability 2^{-g}. An attacker could exploit this even more if the protocol tolerates a specified threshold of errors. Guessing attacks are not only applicable to this specific protocol. A prover executing a timed authentication protocol might be able to implement distance fraud by guessing the last g bits of the challenge before they arrive and start to calculate the response earlier, which provides it with a timing advantage of $g \cdot t_B$ with a success probability of 2^{-g} .

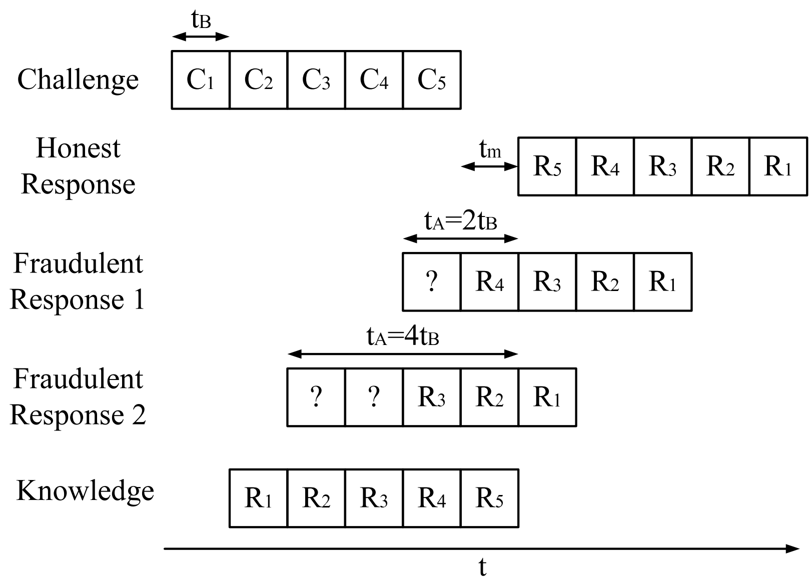


Figure 5.18: Guessing attack on nonce exchange where first response bit depends on last challenge bit. An attacker gains $2t_\mathrm{B}$ for each bit it guesses.

The transmission time for full data packets over normal communication layers and the processing delay of calculating the response prevent protocols from achieving the timing accuracy required. It is important that the processing time t_d is minimized to reduce the uncertainty of the distance-bounding process. The prover therefore needs to calculate possible response strings before starting the exchange stage and the function $C \rightarrow R$ should be simple to implement to minimize t_d further. The exchange of multi-bit data packets over normal communication layers introduces timing latency that can be exploited by an attacker. As a result, protocols specifically designed for distance-bounding applications should rather exchange single bits.

Bit errors

Several protocol proposals fail in the event of a bit error. Some protocols in Sections 5.4.1 to 5.4.3 will fail during the verification stage if the value of bits in the received challenge, or in the received response, are not as expected. A rapid exchange channel, with minimal latency needed for accurate distance bounding, has no error correction and might be implemented on resource constrained devices that contain simple radio receivers more susceptible to noise, so bit errors could occur. It is therefore an advantage if a distance-bounding protocol is resistant to bit errors during the exchange stage.

The protocol in Figure 5.11 handles errors by defining a bit-error threshold. As was pointed out in [64], some protocols in Section 5.4.2 can tolerate bit errors during the exchange stage as long as the challenge bits, C'_i, received by the verifier and the response bits, R_i, sent by the prover are transmitted over an error-corrected channel during the verification stage. For example, the protocol by Brands and Chaum, shown in Figure 5.7, could be altered so that the prover transmits $C', R, (C', R)_{\mathrm{sign}_A}$ during the verification stage. The verifier can then accept the distance bound if at least k_1 of the response bits he received equals those sent by the prover and at least k_2 of the challenge bits the prover received equals those the verifier sent, where $k_1, k_2 > \frac{l}{2}$ and l are security parameters.

An alternative is to use error-correcting codes (ECC). Singelée and Preneel proposed error correction codes to make the MAD protocol, described in Section 5.4.2, resistant to bit errors [149]. ECC $(l + k, l)$ is applied to the response strings M^A and M^B. This results in new response strings $M_1^A, \ldots, M_{l+k}^A$ and $M_1^B, \ldots, M_{l+k}^B$. The protocol now continues exactly as before except that there are now $l + k$ timed bit exchanges instead of l. After the bit exchanges, A reconstructs M^B from the challenges it sent and the responses it received, while B reconstructs M^A in a similar same way. Finally, A uses the ECC to correct any bit errors in the reconstructed M^B and B uses the same code to correct any errors in the reconstructed M^A. The verification stage then proceeds as specified in the original MAD protocol.

5.4.5 Principles for secure distance-bounding communication

To protect against attacks proposed in this chapter, and the attacks proposed in Chapter 6, the designer of a distance-bounding protocol should optimize the choice of communication medium and transmission format according to the following principles:

- **Principle 1**: Use a communication medium with a propagation speed as close as possible to the speed of light in vacuum, which is the physical limit for propagating information through space-time.
- **Principle 2**: Use a communication format in which the recipient can instantly react on the reception of each individual bit. This excludes most traditional byte- or block-based communication formats, and in particular any form of redundancy such as error-correction and packet delimiters such as headers and trailers.
- **Principle 3**: Minimize the length of the symbol used to represent each single bit, in other words output the energy that distinguishes the two possible transmitted bit values within as short a time as is feasible. Alternatively, if working with a baseband signal the verifier should sample as early as possible during the bit period and base his decoding decision on the value of this single sample. This leaves the attacker little room to shorten this time interval further.
- **Principle 4**: The distance-bounding protocol should be designed to cope well with substantial bit error rates during the rapid single-bit exchange, because the previous criterion may limit the reliability of the communication channel.

5.5 A distance-bounding protocol for RFID

Attackers can circumvent the limited range of the near-field channel by means of a relay attack, as demonstrated in Chapter 4. Current RFID tokens can therefore not be used as a reliable method to determine the proximity of a user. Relay attacks cannot easily be prevented by cryptographic protocols that operate at the application layer of an RFID protocol stack. An attacker simply needs to relay the communication between the verifier and the prover for the duration of the transaction. The attacker never needs to know the plain-text data or any key material and as a result the success of the attacker does not rely on a weakness in a specific protocol or algorithm. It is therefore irrelevant whether the verifier authenticates the prover cryptographically or encrypts the data. The only effective defense is a distance-bounding protocol that is integrated into the physical communication layer.

In this section, I discuss in detail the distance-bounding protocol briefly mentioned in Section 5.4.3. This protocol, as shown in Figure 5.19, was specifically designed for use within the RFID environment [64]. A distance-bounding protocol for RFID systems should take into account the limited resources of the token, e.g. no trusted clock and possible errors when processing high-bandwidth communication, and the speed at which a transaction must be executed. The work is based on a protocol idea by Kuhn, who initially proposed the basic relay-resistant protocol that used pre-computation and multiple single-bit exchanges where the responses is selected from a 1-bit lookup table based on the challenge received. Kuhn and I subsequently worked on refining the protocol together. My main contribution was to assist in tailoring the protocol to allow for practical limitations within the RFID environment, to provide a formal definition of the possible threats and to investigate the performance of the protocol when considering bit errors and execution time.

The proposal is based on the following assumptions:

- **Security target**: The protocol only proves to the verifier that the prover is located within a specified distance from the verifier. The protocol will not help to prove this fact to any third party, in other words, it does not provide non-repudiation of location for anyone who does not trust the verifier. The protocol aims to prevent distance fraud and relay attacks. The protocol does not explicitly penalise a prover colluding with an attacker and it does therefore not aim to discourage the "terrorist attack".

- **Cryptographic primitives**: For the purpose of running a distance-bounding protocol, the prover and the verifier share a dedicated secret pseudorandom function (or in practice a dedicated shared secret key and a keyed public pseudorandom function). It is used to calculate the prover's response to a challenge. We assume that the attacker has no access to the shared key or function other than through the radio interface. Our protocol does not depend on any public-key primitives, but should these be available, they can be used to set up the shared key mentioned above before the distance-bounding protocol is initiated.

- **Time base**: The RFID token (prover) is computationally weak. It can compute the secret pseudorandom function mentioned above, but the time it takes for this computation, i.e. several milliseconds, is many orders of magnitude longer than the maximum response-delay variance acceptable for distance-bounding applications, i.e. several nanoseconds. Even worse, the cryptographic calculation progresses according to an externally supplied, and therefore untrusted, clock signal, which a proxy RFID reader (malicious third party) might accelerate in the hope of getting a faster response from the token. We assume that the RFID token has no built-in high-precision time base, e.g. a crystal oscillator. But we do assume that it is reliably able to detect large deviations from its nominal clock frequency, in particular any attempt by an attacker to operate the RFID token at at least *twice* its normal speed (overclocking attack). A simple analog band-pass filter applied to the clock signal can act as a crude trusted time reference, able to prevent a factor-two deviation of the clock frequency. In fact, the tuned resonant circuits in many existing RFID systems, where the carrier frequency is the clock signal, already act as such band-pass filters.

- **Communication channel**: RFID communication channels are not suitable for accurate distance estimation. For distance-bounding, the exchange stage would be implemented on a 'fast' channel, as described in Section 6.3.3, which would provide the required timing resolution. Since this 'fast' channel is implemented on a low-resource device, it is likely that errors will occur during the exchange stage, so our protocol must be resistant to bit errors. Conventional RFID communication, or the 'slow' channel, is used for data transfer during the setup and verification stage and is assumed to be error free. RFID transactions need to be completed as fast as possible, so the protocol must keep data transfer on the 'slow' channel to a minimum.

5.5.1 Protocol description

During the setup stage of our protocol, the verifier V (an RFID reader) sends a nonce N_V to the prover P (an RFID token). N_V is an unpredictable bit string that will never again be used for the same purpose:

$$V \rightarrow P : N_V$$

Both the prover and the verifier then use the pseudorandom function h and the secret key K in order to calculate two l-bit strings R^0 and R^1:

$$R_1^0 R_2^0 R_3^0 \ldots R_l^0 \ || \ R_1^1 R_2^1 R_3^1 \ldots R_l^1 \ := \ h(K, N_V)$$

The function h could be some form of one-way and collision-resistant hash function. The lengths of K, N_V, as well as l, are all security parameters. Typical values should be comparable to the lengths acceptable for symmetric-cryptography keys (80 to 256 bits). If much higher values of l are needed, for example to cope with transmission errors, $R^0||R^1$ can be generated using a secure pseudo random-bit generator seeded from the output of h. In a practical hardware implementation, R^0 and R^1 could be loaded into two shift registers, which are both clocked well before another C_i is received. Instead of using shift registers, the next pair (R_i^0, R_i^1) can also be computed by iterating a pseudo random-bit generator before C_i is received. As the signal propagation time t_p is very small, it is important that the processing delay t_d of the token is short and predictable. From equation 5.1 we can see that variations in t_d greatly effect d, especially if $t_\mathrm{p} < t_\mathrm{d}$. The verifier allows the token a preagreed number of clock cycles to calculate $h(K, N_V)$ and store the result. This effectively separates the cryptographic processing delay from the distance-bounding process, as t_d is reduced to the time it takes an asynchronous digital circuit to lookup and transmit one bit.

The exchange stage consists of l single-bit challenge-response exchanges. The verifier V sends a random challenge bit C_i and the prover P replies instantly with a 1-bit response that is either R_i^0 or R_i^1, selected by the value of C_i. It discards the non-selected value securely at the same time. For all $1 \leq i \leq l$:

$$V \rightarrow P : C_i \in \{0, 1\}$$

$$P \rightarrow V : R_i \in \{0, 1\}$$

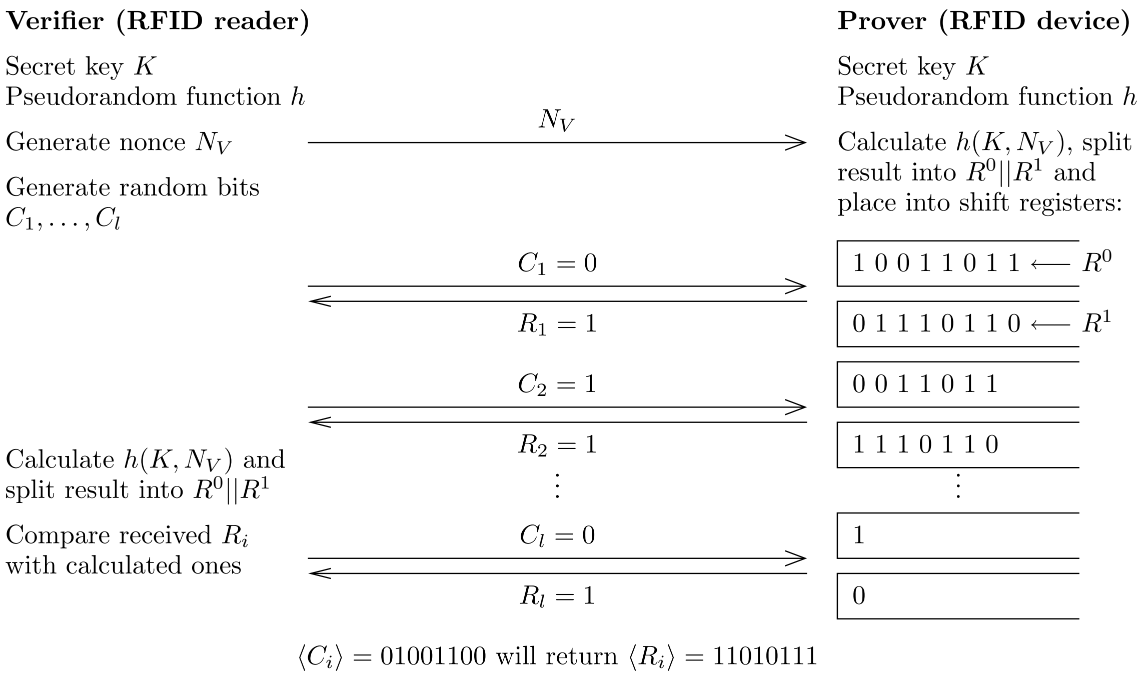

Verifier (RFID reader)
Secret key K
Pseudorandom function h
Generate nonce N_V
Generate random bits $C_1, \dots, C_l$
Calculate $h(K, N_V)$ and split result into $R^0||R^1$
Compare received R_i with calculated ones
Prover (RFID device)
Secret key K
Pseudorandom function h
Calculate $h(K, N_V)$, split result into $R^0||R^1$ and place into shift registers:
N_V
$C_1 = 0$
$R_1 = 1$
$C_2 = 1$
$R_2 = 1$
$C_l = 0$
$R_l = 1$
1 0 0 1 1 0 1 1 ⟵ R^0
0 1 1 1 0 1 1 0 ⟵ R^1
0 0 1 1 0 1 1
1 1 1 0 1 1 0
1
0
$\langle C_i \rangle = 01001100$ will return $\langle R_i \rangle = 11010111$

Figure 5.19: A distance-bounding protocol for RFID

Formulating the response R_i based on the received C_i only needs a single-bit lookup in a 2-bit memory, which can be implemented in an entirely asynchronous fashion, requiring only a small number of gate delays, without any clock signals that the attacker could accelerate to obtain $R_i^{C_i}$ prematurely. If the correct response R_i is received within a sufficiently short time t_m after C_i had been sent out, for each $1 \leq i \leq l$, then using equation 5.1 the verifier is satisfied that the prover is not a distance larger than d away.

The protocol has no data transfer during the verification stage. The verifier checks whether at least k of the l R'_i bits that it received match the expected R_i values it calculated earlier. The value of k is a security parameter and should be chosen to ensure acceptable false acceptance and false rejection probabilities. I discuss k and the protocol's resistance to bit errors further in Section 5.5.3.

5.5.2 Possible threats

Our protocol needs to be resistant to relay attacks and distance fraud. We consider four different threat scenarios and evaluate the security performance of our protocol in each case. A value shown in green is either known or guessed correctly, while a value shown in red is unknown or guessed incorrectly. In the first scenario a third party attacker tries to simply guess the correct responses without interacting with the prover. Figure 5.20 shows an example of this attack. The attacker does not know the key K or the values of R^0 or R^1. The attacker has a $(\frac{1}{2})^l$ probability of answering all l challenges correctly.

In the second scenario a prover tries to commit distance fraud by preemptively transmitting responses. Even though the prover knows the key K and all the values of R^0 or R^1 it does not know the values of $C_1 \dots C_l$. The attacker has a $(\frac{1}{2})^l$ probability of answering all l challenges correctly.

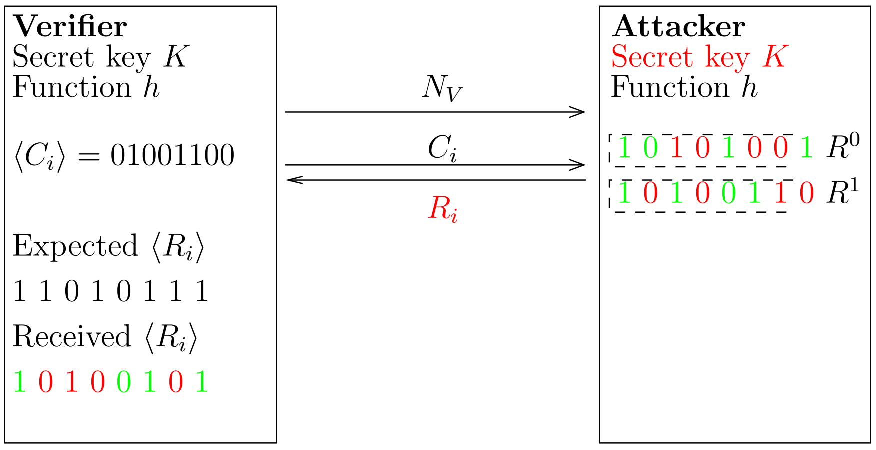


Figure 5.20: Random guess by third party attacker

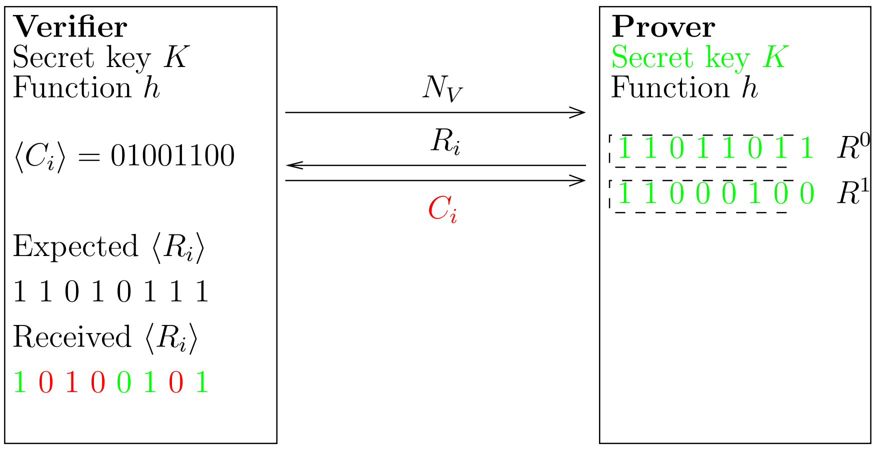


Figure 5.21: Distance fraud by prover

In the third scenario a third party attacker accelerates the clock signal provided to the prover slightly and then transmits an anticipated challenge C_i^A before the verifier reveals its challenge C_i. In half of all cases, the attacker guesses the challenge bit correctly and obtains the correct reply R_i in advance. If $C_i^A \neq C_i$, the attacker will have destroyed the correct reply R_i. In this case the attacker will have to guess the response bit, which he will get correct in half of all cases. Therefore, for each challenge C_i, the attacker has a $\frac{3}{4}$ probability of replying correctly. Overall, the attacker has only a $(\frac{3}{4})^l$ probability of answering all l challenges correctly. This attack is shown in Figure 5.22.

Under normal circumstances, the prover will only ever reveal half of all the bits that it derived from the nonce N_V and the key K. The fourth scenario considers an attacker who tries to retrieve all bits R_i^0 and R_i^1 in advance by attempting to run the protocol twice with the prover (within the time allowed by the verifier for a single run), as shown in Figure 5.23. In both protocol runs the attacker would forward the same nonce N_V, but the values C_i^A in the second run would be complementary to those in the first run. We assumed that this is not possible because the prover has access to a crude trusted time reference that keeps it from running at twice the normal clock frequency. Where this assumption is not practical, the protocol can be modified by adding a prover-generated

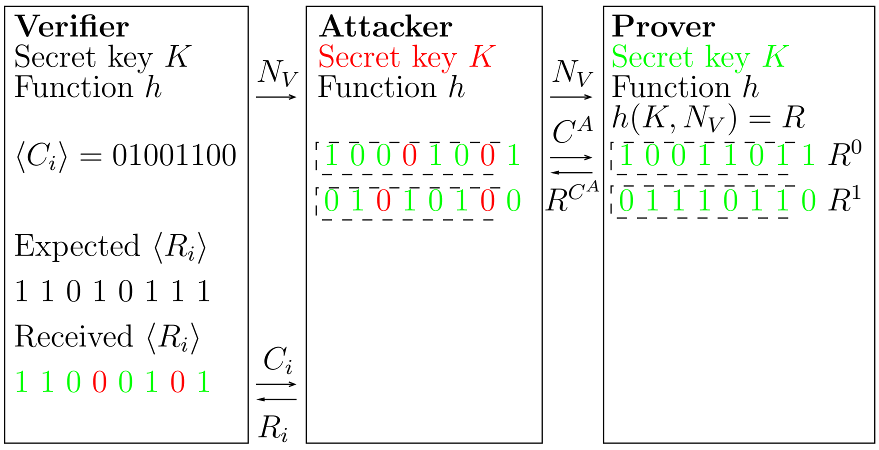


Figure 5.22: Relay attack by third party attacker

nonce. The protocol then starts with both sides transmitting to each other their nonce

$$V \rightarrow P : N_V$$
$$P \rightarrow V : N_P$$

(in any order) and then continues to generate R^0 and R^1 from $h(K, N_V, N_P)$. Neither N_V nor N_P need to be unpredictable to the attacker. They merely must be bit strings that are guaranteed to never repeat during the lifetime of the verifier or prover. In practice, such nonces can either be sufficiently long random-bit strings or they can be strictly monotonic counter values or timestamps. An implementor can choose between the need for including a clock-frequency limiter, some non-volatile memory, a hardware random-bit generator, or a continuously running clock into the prover.

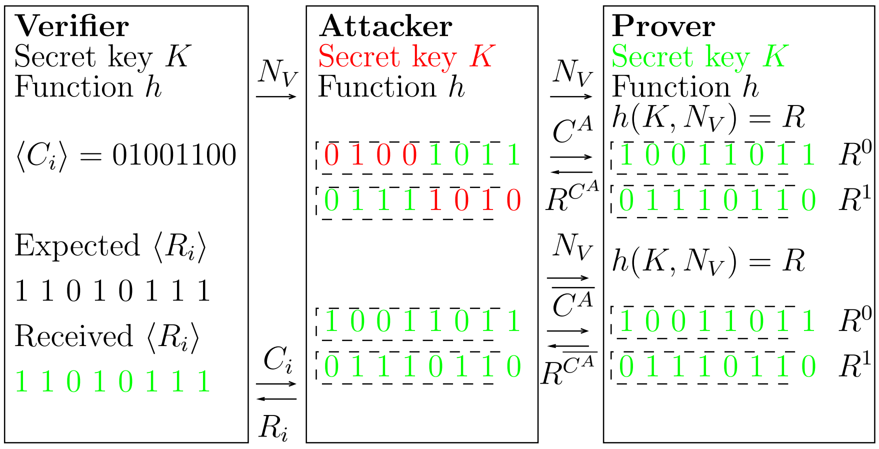


Figure 5.23: Overclocking attack by third party attacker

5.5.3 Execution time and resistance to bit errors

This protocol is ideal for a resource constrained prover as it only needs to evaluate a pseudo-random function and implement a fast asynchronous bit exchange. The prover requires no 'slow' channel to the verifier since it only needs to receive a random bitstring during the setup stage. There is no data exchanged during the verification stage. Our protocol has two main advantages when compared to other proposals: resistance to bit errors and fast execution.

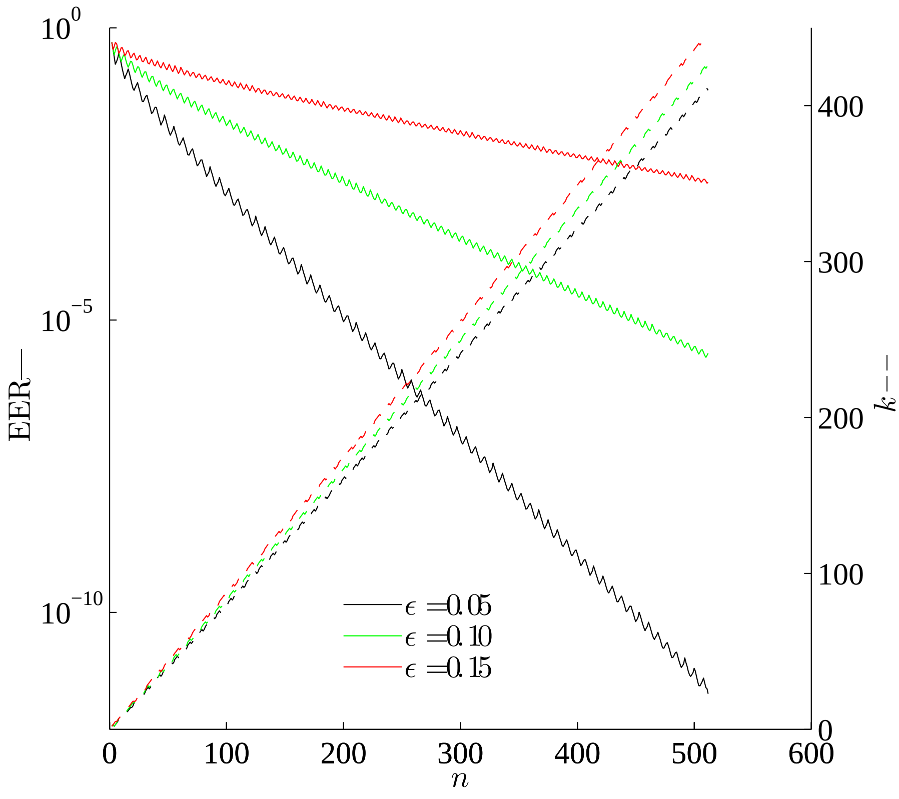


Figure 5.24: An example of parameter tradeoffs in the presence of noise. An implementor knows the bit-error rate ϵ of his channel and chooses the required EER for his application. He then reads off the value for l, corresponding to the required EER on the left axis, and uses it to determine the required value for k on the right axis.

As mentioned in Section 5.4.4 some proposals fail when bit errors occur during the exchange stage. When implementing a reliable rapid-exchange channel on devices with limited resources, many of the sampled C_i or R_i bits may be corrupted. The verifier must therefore be able to accept a prover even when bit errors occur. We propose that a verifier accepts a prover as valid if out of l received R_i bits at least k were correct. An attacker wishing to commit distance fraud, as per scenario one and two in Section 5.5.2, can guess at least k out of the l response bits R_i right with the false-accept probability

$$p_{\mathrm{FA}} = \sum_{i=k}^{l} \binom{l}{i} \cdot \left(\frac{1}{2}\right)^{l} \tag{5.4}$$

A third party attacker that preemptively challenges the prover and retrieves some valid bits from R^0 and R^1, as per scenario three in Section 5.5.2, can guess at least k out of the l response bits R_i right with the false-accept probability

$$p_{\mathrm{FA}} = \sum_{i=k}^{l} \binom{l}{i} \cdot \left(\frac{3}{4}\right)^{i} \cdot \left(\frac{1}{4}\right)^{l-i} \tag{5.5}$$

On the other hand, if ϵ is the probability that a received R_i is corrupted by noise, then the false-reject probability for an honest prover is

$$p_{\mathrm{FR}} = \sum_{i=0}^{k-1} \binom{l}{i} \cdot (1-\epsilon)^i \cdot \epsilon^{l-i} \tag{5.6}$$

The number of transmitted pulses l and the threshold k are security parameters that must be chosen suitably to keep both p_{FA} and p_{FR} within acceptable margins. This choice is also influenced by ϵ. If ϵ and l are kept constant p_{FR} will increase with an increasing k, while p_{FA} would decrease. For each pair (l,ϵ) a value k exists where EER $= p_{\mathrm{FR}} \approx p_{\mathrm{FA}}$. Figure 5.24 shows the tradeoff between the Equal Error Rate (EER), l and k for different values of ϵ.

In a modified Brands-Chaum protocol the prover transmits the C' received and R sent to the verifier. The verifier can then accept the response if the received bit R'_i is equal to $C_i \oplus M_i$ for at least k_1 bits and $C'_i = C_i$ for at least k_2 bits. l is a security parameter and the thresholds must satisfy the condition $k_1, k_2 > \frac{l}{2}$. The worst-case false-acceptance rate for this scheme, even in the presence of a third party attacker, is the same as the best case for our protocol, as shown in Equation 5.4. For the same number of bit exchanges Brands-Chaum therefore offer a much lower p_{FA}. In our protocol, however, we can increase the number of exchanges without much time penalty to achieve the same EER as the modified Brands-Chaum protocol.

For the error-resistant Brands-Chaum protocol the verifier and prover need to exchange l bits on the 'fast' channel and the prover needs to send the verifier $2l$ bits, the R and C' strings, over the 'slow' channel during the verification stage. If we assume that the 'fast' channel exchanges x bits per second and the 'slow' channel transmits y bits per second the time taken to complete this transmission, not taking into account the transmission of additional commitment and verification data, is $\frac{2l}{y} + \frac{l}{x}$ seconds. Our protocol only needs to exchange l bits over the 'fast' channel so the additional transmission time, not including the nonce exchange, is $\frac{l}{x}$ seconds. Tables 5.1 and 5.2 show comparative bit transmission times for these two protocols when implemented on standard RFID tokens. These examples show that our protocol can achieve the same EER by increasing l and still execute faster.

ISO Standard	Time (B and C) $l = 70$	Time (Our protocol) $l = 360$
15693 'fast' 26.4 kbit/s, 13.56 MHz	5.373 ms	0.36 ms
15693 'long' 6.62 kbit/s, 13.56 MHz	21.218 ms	0.36 ms
14443 A/B 106 kbit/s, 13.56 MHz	1.391 ms	0.36 ms

Table 5.1: Time needed to exchange the required data to achieve EER $= 10^{-4}$ and $\epsilon = 0.1$ Assume fast bit exchange rate is 1 Mbit/s

In 2007 Singelée and Preneel proposed error correction codes (ECC) to make the MAD protocol, as shown in Figure 5.8, resistant to bit errors [149]. The basic idea is to create a response string, $M_1, \ldots, M_k$, and challenge, $C_1, \ldots, C_k$, to which is applied ECC(l,k). The

ISO Standard	Time (B and C) $l = 125$	Time (Our protocol) $l = 440$
15693 'fast' 26.4 kbit/s, 13.56 MHz	9.594 ms	0.44 ms
15693 'long' 6.62 kbit/s, 13.56 MHz	37.890 ms	0.44 ms
14443 A/B 106 kbit/s, 13.56 MHz	2.484 ms	0.44 ms

Table 5.2: Time needed to exchange the required data to achieve EER $= 10^{-10}$ and $\epsilon = 0.05$. Assume fast bit exchange rate is 1 Mbit/s

ECC appends $l-k$ bits to the response string and the challenge. The protocol executes the rapid bit exchange stage using $M_1, \ldots, M_l$, and challenge $C_1, \ldots, C_l$. The $l-k$ redundant bits are then used to correct up to x errors in the first k bits during the verification stage. These redundant bits do not offer any extra security since they can be calculated from the first k bits. An attacker cannot disguise a wrong guess as a bit error since he will have to apply ECC (l,k) to his response. The verifier will not be expecting any bit errors after correcting the response, so it will reject the prover if any of the guessed bits are incorrect. The authors therefore claim that an attacker committing distance fraud has to guess the first k bits correctly and that

$$p_{\mathrm{FA}} = \left(\frac{1}{2}\right)^k \tag{5.7}$$

The authors claim that the ECC method works better than the error threshold we propose. They, however, compare their best case, against distance fraud, to our worst case as shown in Equation 5.5, which is against a relay attacker that challenges the prover early to retrieve a portion of the response strings before the exchange stage starts. They do not consider the performance of the ECC scheme against a relay attack. An attacker can implement the relay as follows: First he generates a k-bit response string M'. The attacker waits for the exchange stage to start and responds with bit $R'_i = M'_i \oplus C_i$ to the verifier's challenge. He still relays the challenge C_i to the real prover and subsequently also receives the correct responses R_i back. After k bit exchanges the attacker knows the prover's response string $M_i = C_i \oplus R_i$ for $i = 1, \ldots, k$. As a result, the attacker can now calculate the last $l-k$ bits based on the first k bits of M he has discovered. The only slight delay might be on bit R_{k+1} while the attacker calculates ECC (l,k). Using the correct redundant bits, the verifier will apply the ECC to $M' = C \oplus R'$ expecting to recover M. The attacker would therefore succeed if less than x bits of M' differ from M, since any incorrect bits in M', and by implication R', would be interpreted as bit errors. The false acceptance rate for the ECC against a relay attack is therefore

$$p_{\mathrm{FA}} = \sum_{i=k-x}^{k} \binom{k}{i} \cdot \left(\frac{1}{2}\right)^k \tag{5.8}$$

From the examples in Tables 5.3 and 5.4 it can be seen that Brands-Chaum, with a specified threshold, is the best error resistant protocol in terms of false acceptance probability if the designer is willing to incur an execution time penalty. The ECC method is slightly better in terms of false acceptance probability for lower bit-errors but significantly worse when x increases. The prover, however, also requires additional resources to implement

	Hancke-Kuhn		ECC		Brands-Chaum	
# allowed errors x	k	p_{FA}	ECC(l,k)	p_{FA}	k	p_{FA}
$x = 7$	30	0.259	(37,9)	0.981	30	9.554×10^{-5}
$x = 4$	33	0.028	(37,16)	0.038	33	5.422×10^{-7}
$x = 3$	34	0.009	(37,22)	4.278×10^{-4}	34	6.166×10^{-8}
$x = 2$	35	0.002	(37,26)	5.245×10^{-6}	35	5.122×10^{-9}
$x = 1$	36	3.178×10^{-4}	(37,31)	1.490×10^{-8}	36	2.765×10^{-10}
$x = 0$	37	2.384×10^{-5}	(37,37)	7.276×10^{-12}	37	7.276×10^{-12}

Table 5.3: Performance of Hancke-Kuhn threshold, ECC and Brands-Chaum threshold when $l \approx 37$

	Hancke-Kuhn		ECC		Brands-Chaum	
# allowed errors x	k	p_{FA}	ECC(l,k)	p_{FA}	k	p_{FA}
$x = 13$	50	0.261	(63,12)	1.000	50	1.508×10^{-6}
$x = 10$	53	0.058	(63,18)	0.760	53	1.691×10^{-8}
$x = 7$	56	0.005	(63,28)	0.006	56	6.818×10^{-11}
$x = 5$	58	5.111×10^{-4}	(63,37)	3.714×10^{-6}	58	8.312×10^{-13}
$x = 3$	60	2.300×10^{-5}	(63,47)	1.232×10^{-10}	60	4.524×10^{-15}
$x = 1$	62	2.960×10^{-7}	(63,57)	4.025×10^{-16}	62	6.939×10^{-18}

Table 5.4: Performance of Hancke-Kuhn threshold, ECC and Brands-Chaum threshold when $l \approx 63$

the chosen ECC and the number of allowed errors are fixed, whereas our protocol is flexible with the verifier having the option to change the security parameters 'on-the-fly' if the characteristics of the communication channel changes.

5.6 Conclusion

In this chapter, I discussed the possibility of proving physical proximity cryptographically by using distance-bounding protocols. Several proposals for distance measurement were discussed and I explained why the time-of-flight (ToF) measurement of RF signals are most suitable to secure distance-bounding. I discussed ways to implement ToF distance-bounding protocols and reviewed a number of published proposals. Several protocol proposals fail if bit errors occur during the exchange stage. Some proposals also fail to consider the latency introduced by processing responses and the underlying communication channel. I explain how an attacker can manipulate the processing time t_{d} to gain a timing advantage. I also show how the attacker can decrease the measured round-trip time for a single multi-bit exchange by guessing and preemptively transmitting response bits or by exploiting communication layer latencies. These attacks can be successfully applied to existing proposals. I proposed a number of principles to adhere to when implementing distance-bounding systems even though these make the practical implementation more challenging. These restrict the choice of communication medium to speed-of-light channels, the communication format to single bit exchanges for timing, symbol length to narrow ultra wideband pulses, and protocols to error-tolerant versions.

Finally I describe a protocol that provides secure distance-bounding for the RFID environment. The protocol protects against relay attacks and ensures that an RFID token is within an acceptable distance from the RFID reader. It requires little processing resources from the prover, with all precision time measurements and random bit string generation being done by the verifier. The prover only needs to execute a pseudo-random function. This protocol could therefore be implemented on RFID tokens ranging from simple tags to contactless smart cards. This protocol can also be extended to applications involving resource constrained devices. The protocol is best used for radio-frequency or optical ranging, which are more suited for security applications than ultrasonic or RSS concepts. The protocol assumes that the prover and verifier has a 'slow' error-corrected channel for data transmission and a 'fast' bit-exchange channel. The proposed protocol only sends one nonce from the verifier to the prover during the setup stage and requires no verification stage. In fact, unless the two-nonce variant is chosen, the protocol does not require any noise-free communication channel from the prover to the verifier. This not only simplifies its implementation, but also makes it suitable for applications where a rapid completion of the protocol is required.

If the bit-exchange channel needs to be implemented on a 'fast' channel, which provides the required timing accuracy, timing bit errors could occur. Communication errors are handled simply by tolerating some bit errors during the single-bit challenge-response exchanges. In an error-free environment, the protocol requires about twice as many single-bit challenge and response exchanges for the same level of security than the Brands-Chaum protocol shown in Section 5.4.2. This is because an attacker can guess a correct response with probability $\frac{3}{4}$ when preemptively challenging the prover, compared to probability $\frac{1}{2}$ with Brands-Chaum. When bit errors do occur, the Brands-Chaum protocols would have to be modified to transmit all l bits of C actually received and the l bits of R actually sent to the verifier. If this is not done, the verification stage will fail, because the prover will not be able to verify the signature correctly, since his version of m will differ from the prover's version of m. This additional transmission not only doubles the number of bits needed by the Brands-Chaum protocol, but this additional data have to go over the reliable 'slow' channel, along with the additional data from the setup and verification stages, which substantially increases the time required to complete the Brands-Chaum protocol. It is also shown that using a threshold method to allow for bit errors results in a lower false-acceptance rate than the MAD protocol using error-correction codes when a higher number if bit errors occur.

Chapter 6

Distance-bounding channels

Distance-bounding protocols make distance estimates based on the round-trip time of a cryptographic challenge-response exchange. The exchange stage must therefore be implemented using a communication channel that will provide an accurate and secure time measurement. In this chapter I investigate whether conventional RF communication channels are appropriate for use with distance-bounding protocols. I also discuss proposals for distance-bounding channels and describe the implementation of a rapid bit-exchange channel for near-field devices.

6.1 Introduction

Time-of-flight distance-bounding protocols must be integrated into the physical layer of the communication channel to accurately determine 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. Ultrasound, for example, is not a good channel medium since the propagation speed is much slower than that of radio waves, leaving the system vulnerable to relay attacks. The implementation of a suitable exchange channel is not often discussed in literature, despite its importance.

In Chapter 5, I described several distance-bounding proposals and discussed weaknesses at the packet level, i.e. data format, of the communication channel. In this chapter I discuss how the physical communication layer, i.e. coding and modulation, can be exploited by the attacker to gain a time advantage. I argue that conventional RF channels, often found in RFID and sensor networks, are not suitable for implementing secure distance-bounding and I show how an attacker can practically implement 'late-commit' and overclocking attacks against current RF receiver architectures. In light of these weaknesses, I look at existing proposals for distance-bounding and relay-resistant communication channels and comment on their effectiveness. I also describe the design and possible implementation of a distance-bounding channel for near-field devices.

This chapter is based on work I did for a paper co-authored with Kuhn, "Attacks on 'Time-of-Flight' Distance Bounding Channels" [63]. This chapter also contains work from a paper I co-authored with Kuhn in 2005, “An RFID Distance-Bounding Protocol” [64]. The idea of a carrier-synchronised distance-bounding channel for RFID devices was first proposed by my co-author. I subsequently worked on the practical implementation of such a channel for near-field devices.

6.2 Attacks on the physical communication layer

Distance-bounding protocols require accurate timing in order to estimate the Round-Trip-Time (RTT) of challenge-response pairs. Let us consider a simple system where the verifier starts a timer when it sends a challenge bit and stops the timer when it detects the start of a response bit sent by the prover. If all system components have a predictable time delay, the verifier now has a good RTT estimate and can therefore calculate an upper bound on the distance to the prover. Accurate timing alone does not, however, ensure that the protocol is secure, as the actual response value still needs to be determined. If an attacker could start a response within the allowable time period but still change the value at a later stage, once he has observed the prover’s response, the protocol’s security is compromised. The verifier must therefore ensure that the prover commits to the response value at a well-defined point of time, which effectively links the time measurement with the cryptographic exchange.

Kuhn first introduced the idea of ‘Deferred Bit Signaling’ [35], where an attacker could attack a receiver that integrates the signal energy over an entire bit period. The attacker would send no energy for $\frac{m-1}{m}$ of the time interval and then send a signal, amplified m times, during the final $\frac{1}{m}$ of the time interval. The result of the receiver’s integration would be the same, but the attacker can delay committing to a bit’s value by $\frac{m-1}{m}$ of the bit period. This notion that an attacker can change a bit’s value after its transmission time has begun has served as a starting point for the work presented in this section. I build on this idea and show that late-commit, or late-send, attacks can be implemented in a number of ways by exploiting features of the receiver architecture during decoding and demodulation.

The late-commit attack can be used in both distance fraud and relay attacks. A fraudulent prover can commit distance fraud by preemptively guessing a response and then, if required, changing it to the correct value once the challenge is received. In relay attacks the attacker cannot avoid introducing a delay when relaying the challenges and responses, which could cause the round-trip-time to exceed the limit set by the verifier. The attacker can use the late-commit attack to guess the prover’s response and then, if needed, change his guess once he receives the actual response from the prover. In this case an attacker would implement a special receiver that determines the response of the prover early in the bit period, which still gives him time to alter his response. This technique can also be used to ‘shorten’ the time taken to relay the challenge from the verifier to the prover. An example of this type of relay attack is shown in Figure 6.1.

6.2.1 The communication channel

A typical communication channel consists of a transmitter sending data to a receiver using an RF carrier. Data sent over this channel must first be encoded and then modulated onto the carrier. Coding changes the binary data into a signal sequence that is best suited to the

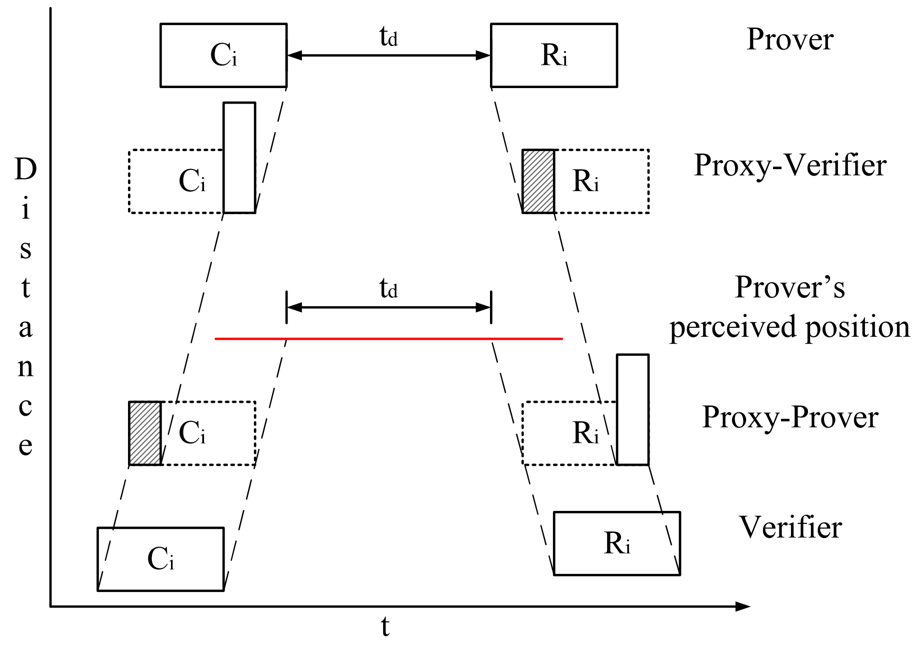


Figure 6.1: In this variation of the relay attack, the attacker gains time when the proxy prover estimates the value of the challenge bit from the verifier early on in the bit period, and the proxy verifier transmits m times the symbol amplitude to the prover in the final $\frac{1}{m}$-th of the bit period. The process is then repeated for the response bit, albeit with the proxy verifier and prover swapping roles.

transmission channel and aids the receiver in recovering the data, e.g. Non-Return-to-Zero (NRZ), Manchester, etc. Modulation is the process by which the amplitude, frequency and phase of an RF carrier is altered in relation to the resultant baseband signal. The receiver must then perform demodulation and decoding to recover the data, as shown in Figure 6.2.

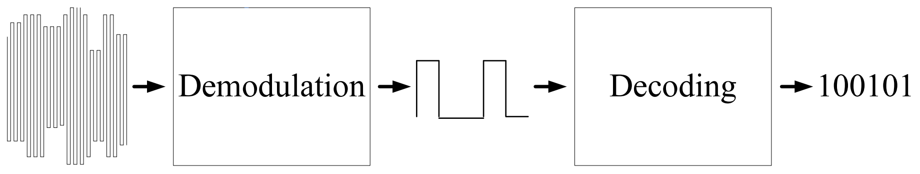


Figure 6.2: Data recovery at the receiver

Super-heterodyne receivers

Sensor nodes generally use RF carriers in the ISM bands (315 MHz, 433 MHz and 2.4 GHz) with simple modulation schemes such as Amplitude-Shift Keying (ASK) or Frequency-Shift Keying (FSK). For example, the popular Mica2 node by Crossbow Technology [39] uses FSK at 315/433 MHz. Most of the RF receivers, including the Chipcon CC1000 [34] on the Mica2 node, use the super-heterodyne architecture shown in Figure 6.3.

The incoming carrier is mixed down, using a synthesized clock from a phase-locked loop (PLL), to an intermediate frequency band (IF) where it is filtered and amplified. The IF stage often contains a limiter, to prevent saturation in the remaining receiver circuitry. The coded data is demodulated off the IF carrier and quantized by a data slicer. For ASK,

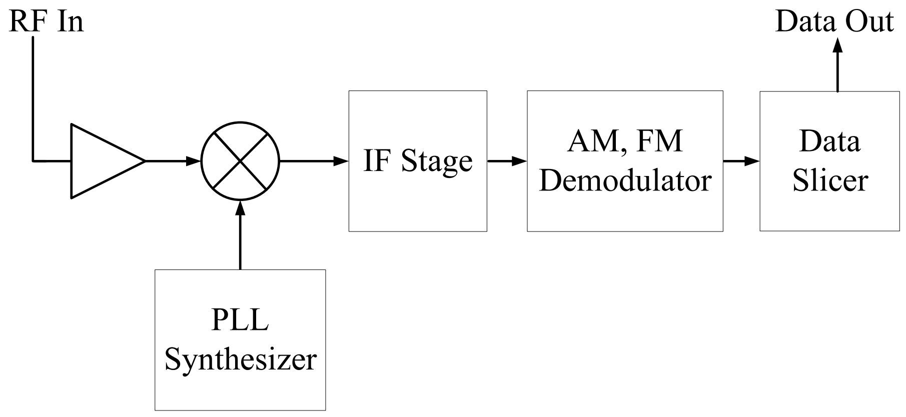


Figure 6.3: Functional diagram of a generic superheterodyne RF receiver

the IF carrier is rectified and passed through a low-pass filter (envelope detector), while the carrier is fed into another PLL for FSK. The data slicer is usually a comparator with a dynamic reference. This stage may include additional low-pass filters to remove high frequency glitches. Some receivers also do decoding, but most of the time this function is performed by another logic device, such as a micro-controller or FPGA connected to the receiver. For the practical work in this paper, I used the MAXIM 1471 433.92 MHz ASK/FSK receiver evaluation board [106] and the RF Solutions RRFQ2 433.92 MHz FM receiver [136].

RFID receivers

RFID tokens used for proximity authentication use a 13.56 MHz carrier with a two-stage modulation process. The coded data is first modulated onto a low-frequency (847 kHz/423 kHz) sub-carrier before being amplitude modulated onto the HF carrier. RFID receivers use an architecture similar to the one shown in Figure 6.4. First the 13.56 MHz carrier is removed. This can be done by rectifying the carrier and passing the result through an envelope detector. Alternatively, a zero-IF system could be implemented, where the received signal is mixed with a 13.56 MHz clock. The signal is then low-pass filtered to leave only the modulated sub-carrier, which is then amplified, demodulated and digitized.

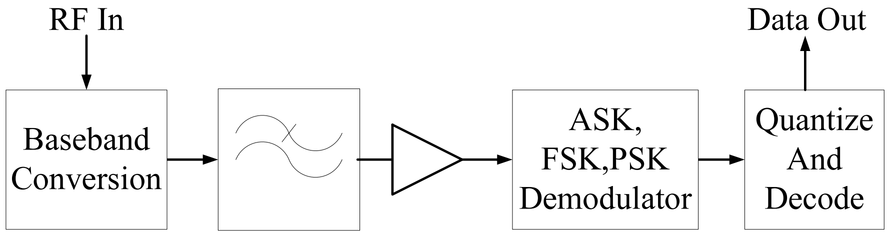


Figure 6.4: Functional diagram of a generic 13.56 MHz RFID receiver

For our practical experiments, I used the NXP MF RC531 Contactless Reader IC [123]. This receiver uses an IQ demodulator to recover the coded data from the sub-carrier and also performs decoding. During the demodulation process the received signal is correlated with an expected base function, which produces a peak in the output whenever the signal corresponds closely to the base function. Since the base function is rectangular in this case, the process simplifies to integrating over a bit period, as explained in Chapter 3, so this receiver architecture is ideal for testing the 'Deferred Bit Signaling' attack. For comparison I also studied the Melexis MLX90121 13.56 MHz RFID transceiver [110]. The receiver's data sheet does not describe how it performs the required sub-carrier demodulation, but it does contain an additional 'majority voting' step to help filter noise and correct distorted signals during digitization. An example of a majority voting scheme is shown in Figure 6.5. Multiple samples are taken over the bit period and at the end a decision is made as to whether the signal should be high or low. This is in effect similar to integrating over the bit period, or averaging, and comparing to a threshold.

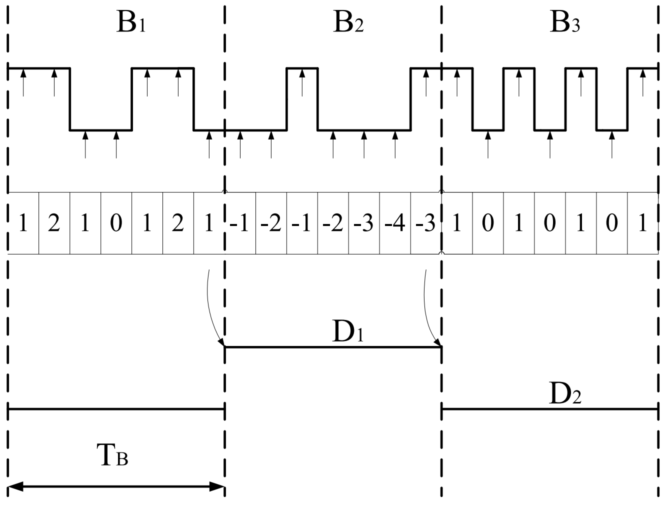


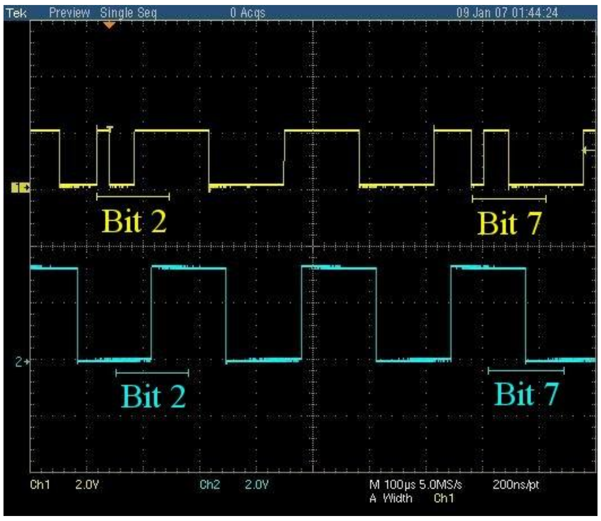
Figure 6.5: Example of majority voting over a bit period. Here the receiver samples the incoming signal $B_{1,2,3}$ seven times during the bit period T_B. It then decides, based on the vote, whether to make the corresponding output $D_{1,2}$ high or low.

6.2.2 Late-commit attacks

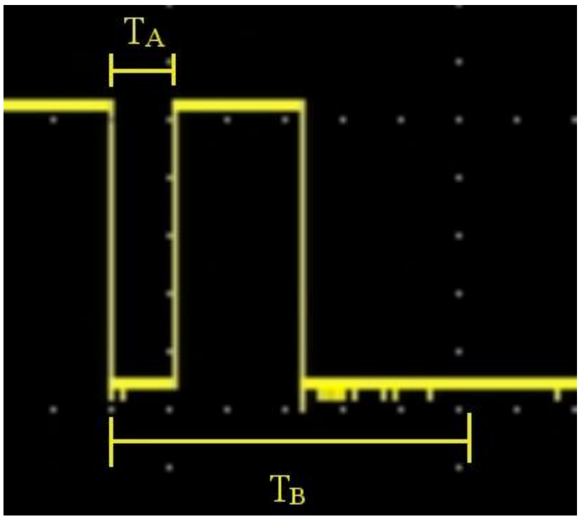
I implemented 'late-commit' attacks against two RF ICs that use the above receiver and decoder structures. In both cases, it was possible to start transmitting a response and then change the value of this response later during T_B. I used two different experimental setups to implement the attack. I then discuss further attack strategies against these and several other commonly used decoding techniques, as well as counter measures and clock attacks.

UHF receiver

In the case of the Maxim 1471 ASK/FSK receiver, I connected an RF signal generator (HP E4421B) directly to the antenna input and provided a 433.92 MHz carrier suitably modulated with a data stream generated by an FPGA board. My aim was to exploit the low-pass data filters in the AM demodulator and data slicer to implement a 'late-commit' attack. These filters are designed to remove unwanted demodulation products, as well as glitches with a shorter duration than the bit period T_B. This, however, means that if the attacker changes his response after a short time period (T_A), his initial incorrect response is filtered out and the receiver outputs the expected data stream.



(a) Bit 2: ‘1’→‘0’, Bit 7: ‘0’→‘1’



(b) Bit 7: $T_A \approx 22$ µs $\overset{\Delta}{\approx}$ 6.6 km (at speed of light)

Figure 6.6: Late-commit attack exploiting the receiver’s data filter.

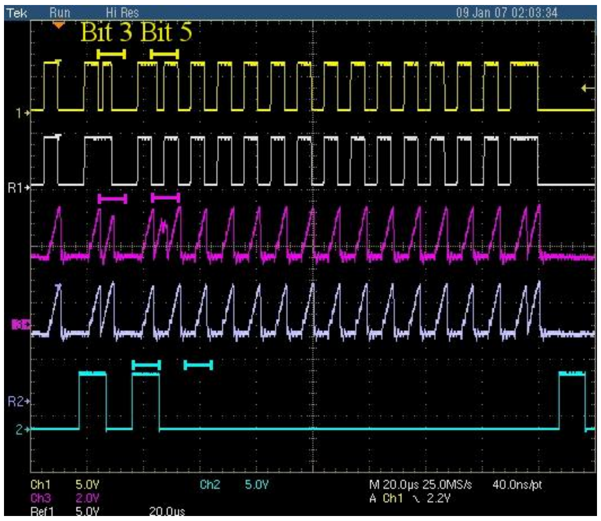
Figure 6.6 shows an example implementation of this attack against the Maxim 1471 receiver. The top waveform, in (a), shows the irregular data stream and the bottom waveform shows the output of the receiver. In the example, a Manchester encoded data stream (10101010) is transmitted and bits 2 and 7 are changed to illustrate the attack. In (b), a magnified trace of bit 7 is shown. The attacker assumes that the next bit will be the inverse of the current bit and starts to transmit the relevant data. He then finds that either he is correct, in which case he keeps with his current value, or that he is wrong and he changes the value accordingly.

In my case an attacker can still commit to the right value $T_{\mathrm{A}} = 22$ µs after he started the response, and his incorrect attempt will be filtered out by the receiver. The demodulated data will be the same as if he guessed the response correctly from the start and the receiver output will be as expected by the verifier.

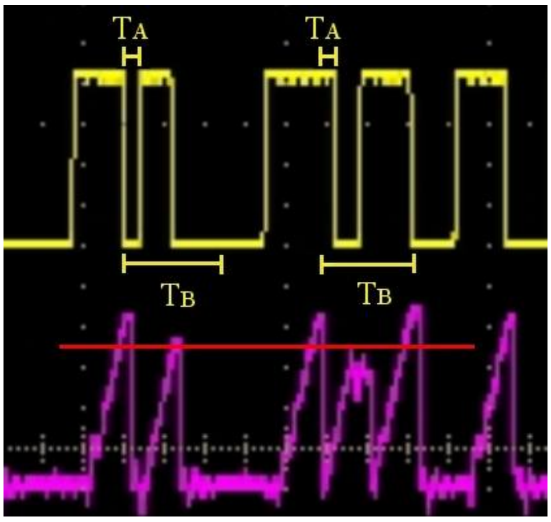
ISO 14443 reader

In the case of the NXP MF RC531 contactless reader IC, I used an ISO 14443A compliant test PICC (proximity integrated circuit card), described in [54], to load modulate the 13.56 MHz carrier with a data stream from the FPGA board. The data steam was formatted according to ISO 14443A, i.e. 106 kbit/s Manchester coded data. The NXP MF RC531 has several debugging outputs that allowed me to observe the signal waveforms at different stages in the receiver.

In this case, my aim was to exploit the integrator to not commit to a bit value at the start of the bit period. The attacker starts to respond with an arbitrary value and then changes it once he knows the correct one. This must happen soon enough to ensure that the correlation result ends up on the correct side of the decision threshold. Figure 6.7 shows an example implementation of this approach where the PICC answers with an ATQA (Answer to Request: Type A) in response to a REQA (Request: Type A) command from the reader. The top trace, in (a), shows the irregular data stream, the second trace shows the correct response, the third trace shows the output of the correlation stage for the irregular input, the fourth trace shows correlation stage output for the correct input and the bottom trace shows the output of the receiver measured for both inputs. In (b), a magnified trace of bit 3 and 5 showing T_{A}, T_{B}, the corresponding correlation output and the threshold is shown. During bit 3 the attacker initially guesses low but then changes to high, still ensuring that the correlation peak is large enough. In bit 5 the attacker guesses high but then changes his answer to low before the correlation peak reaches the threshold. In this case, an attacker can gain almost a quarter of the bit period, approximately 2.5 µs.



(a) Bit 3: ‘0’→‘1’, Bit 5: ‘1’→‘0’



(b) $T_A \approx 2.5$ µs $\overset{\Delta}{\approx}$ 750 m (at speed of light). The red line shows the level of the decision threshold.

Figure 6.7: Late-commit attack exploiting the correlation demodulator in RFID receiver

Strategies

Which signal value an attacker should best transmit initially, before deciding on a bit value, depends on which range of signal values the attacker can achieve at the input of the integrator. If the attacker were able to achieve arbitrarily large positive and negative input voltages there, then the initial voltage would not matter much, as the integration result could be changed in any direction by large values at the last moment.

At the output of an ideal linear AM demodulator, an attacker could achieve arbitrarily large positive voltages, but no negative voltage due to the rectification in an envelope detector. In this case, the attacker would need to start out with zero voltage to keep the accumulating value in the integrator low for as long as possible, because it will be easy to increase the output of the integrator later, but there will be no way to reduce it. This is the scenario that motivated the original ‘Deferred Bit’ attack, in which a ‘1’-bit is represented by the lowest possible base-band voltage for $\frac{m-1}{m}$ of the bit period and by a voltage m-times the one normally used for a ‘1’-bit during the final $\frac{1}{m}$-th part of the period T_{B}.

In practice, the output of a linear or logarithmic demodulator might be limited to voltages in the range 0 to V_{m}, with the binary threshold for the integration result set at value $T_{\mathrm{B}} \cdot V_{\mathrm{m}}/2$. Such limits may either have been introduced intentionally, to protect later circuit components from large amplitude signals, or they may just be an unintended side effect of amplifier and supply-voltage limitations. In either case, the optimal late-commit strategy for an attacker facing two voltage limits 0 and V_m is to initially aim at a demodulator output voltage of $V_\mathrm{m}/2$, and then to switch to 0 or V_m when the desired bit value is known. This keeps the integrator heading for exactly the threshold level, ensuring that even a very brief voltage-limited deviation at the last moment can still move the result to either side, thereby maximising the attacker's timing advantage T_A.

If a constant integrator input voltage that would lead to an integration result identical to the threshold value is not achievable, e.g. because of non-linearities such as threshold elements between the demodulator output and integrator input, then alternating between the voltages corresponding to '0' and '1' during the undecided period might be used to achieve the same effect.

Other decoder algorithms

During the decoding stage, a device needs to decide if a '1' or a '0' was transmitted during a particular bit period T_B. Using an integrator to determine the average input voltage during the bit duration, followed by applying a threshold, is only one of several commonly implemented decoding techniques. Others involve sampling the output of a simpler low-pass filter that crudely approximates the function of the integrator. To prevent bit errors due to clock jitter, these sampling times usually incorporate a safety margin to ensure that the receiver samples not too close to the boundaries between bit periods.

Let us look briefly at popular methods to decode NRZ and Manchester signals. In each case, the device requires a locally generated clock to periodically sample the incoming signal. In the case of NRZ, a common method is to sample once, preferably at $\frac{1}{2}T_B$ after the start of the bit period, during each bit period and assign a '1' to a high, and '0' to a low input state. For Manchester coding, the device might take a sample S_1 at $\frac{1}{4}T_B$ and a sample S_2 at $\frac{3}{4}T_B$. The result S_1 = high and S_2 = low would decode to '1', S_1 = low and S_2 = high would decode to '0', and any other combination would be invalid.

In some cases, devices take more than one sample per bit period and determine the value by a majority voting scheme similar to the one in Figure 6.5. For example, the USART module of a PIC16F876 micro-controller will sample three times during each bit period and make a decision on the number of those samples corresponding to high or low [111].

An attacker could exploit the conservative sampling times during these decoding processes to commit late. The exact attack method and time gain T_A would depend on the specific decoding method and filter time-constants. For example, when the channel uses NRZ encoding and the verifier samples each bit once, the attacker needs to apply the correct bit value only one or two filter time-constants before $\frac{1}{2}T_\mathrm{B}$. If the device samples earlier and more often during the bit period and uses a majority voting scheme, this does not make the situation any worse for the attacker. He can then send balanced data and use the last few samples to move the counter to the desired side of the threshold. Figure 6.8 shows example attacks on NRZ, Manchester and a majority voting scheme, and indicates in each case the time T_A gained by the attacker before he needs to commit to a value.

To exploit such decoding steps, an attacker needs to send pulses shorter than T_B. In the NRZ and Manchester examples, the demodulator and any subsequent smoothing filter need to allow through pulses of somewhat less than half a bit period length, i.e. for the

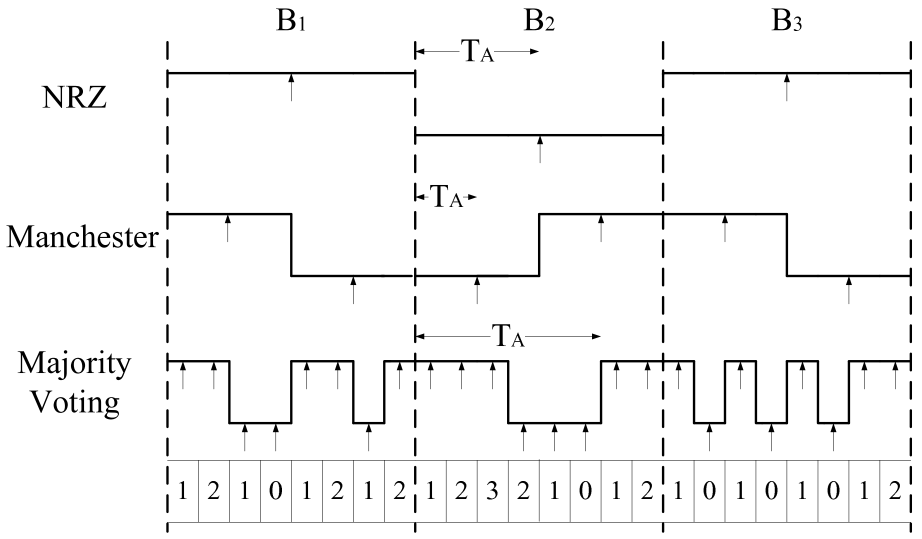


Figure 6.8: Examples of 'late-send' attacks on the decoding stage.

attack to work, the receiver must allow frequency components higher than the intended data frequency to pass. In some cases, nodes communicate with each other using a receiver that can easily transmit higher frequency data, e.g. using a 100 kbit/s receiver to receive 9.6 kbit/s serial data. To test this, we kept the received signal strength constant while increasing the data frequency. We found that the receivers tested reliably demodulate data above the data filter cut-off frequency. At −100 dBm the Maxim and RF Solutions FSK receivers managed to demodulate 15 kbit/s NRZ data even though their low-pass data filters have cut-off frequencies of 9.6 and 4.8 kHz respectively.

6.2.3 Countermeasures

The underlying vulnerability of all attacks presented so far is that the long bit duration T_{B} is substantially longer than the time that light needs to travel twice the distance that marks an acceptable accuracy for a distance-bounding scheme. In the case of a 9.6 kbit/s channel used on a sensor node, this half-bit length is more than 15 km, in the case of a 100 kbit/s RFID channel it is still 1.5 km.

So the obvious countermeasure is to adhere to the four principles for secure time-of-flight distance-bounding in Chapter 5, in particular Principle 3, which states that T_{B} should be as short as possible. It should be noted that this countermeasure does not prevent the attacks but aim to minimize the time T_A gained by the attacker. To accomplish this the receiver architecture and decoding routines would have to be modified to increase the bandwidth and sample, or evaluate, the bit value earlier in the bit period.

It is, however, not always practical for reasons of cost and compatibility to make these modifications to the transmitter and receiver hardware. Nevertheless, in some circumstances it may be feasible to modify a decoding process and switch to an alternate filter only during the execution of a distance-bounding protocol. This particularly applies to the case where a software routine decides on the exact time of sampling the output of a demodulation filter, or even applies a majority vote to several such samples. In this case, the software only has to be modified to sample the values, considered in the bit value decision, as early as possible in the bit period, to the extent allowed by the filter's time constant. This, however, means that only a fraction of the energy normally transmitted for each bit will be utilized to distinguish it from background noise, and as a result, this approach will lead to higher bit error rates. This approach also requires accurate timing and synchronization between the transmitter and receiver, since the idea behind voting or sampling in the middle of the period was to allow for differences and drift in their respective clocks. While the resulting increase in bit error rate may not be tolerable for regular data transmission purposes, it may be sufficient for use with a distance-bounding protocol that was designed for use on highly unreliable channels, such as [64, 149].

Other approaches involve using tighter decision thresholds, in particular the use of separate thresholds for '0' and '1' bits. It is clear from Figure 6.7 that if an attacker incorrectly guesses low he causes the correlation peak to be smaller. An attacker has to ensure that the peak does not fall below the threshold in order to still have that bit period specified as high. Conversely an attacker incorrectly guessing high causes a greater correlation peak and he needs to ensure that this peak does not exceed the threshold. The RFID receiver I looked at allowed the user to set a single threshold level. This is insufficient since setting the threshold high will prevent a decrease in T_A in the case where a high bit was transmitted, but an attacker would have a greater T_A when a low bit is transmitted. The opposite scenario also holds true and although this might be sufficient to detect attacks, the best fix would be to implement two thresholds. A high would then have a correlation peak greater than the first threshold and a low would have a correlation peak less than the second threshold, therefore decreasing T_A in both cases. Two thresholds would also increase the bit-error rate as the thresholds are closer to the expected integration result and the probability increases that additional noise will cause the result to be on the incorrect side.

6.2.4 Clocking attacks

In addition to the late-commit attacks we also consider the possibility that the attacker could speed up the reply from the prover, as previously mentioned in [35]. This attack is especially relevant for protocols that expect the prover to first receive an entire multi-bit challenge before replying. Getting the correct response earlier than expected could allow the attacker enough time to relay the response back to the verifier.

This attack assumes that the sampling, or data, clock is not generated independently by the receiver but recovered from the encoded data. Since the attacker controls the transmission of the encoded data to the prover, he can alter the sequence in such a way as to increase the frequency of the data clock, which would cause the prover to decode the data in a shorter time and reply early. To illustrate this principle I devised a proof-of-concept attack against a Manchester decoder, which is implemented using either a PLL or counter for clock recovery as described in [172].

It is theoretically easier to exploit clock recovery done using a PLL. The PLL locks onto the incoming signal and synthesizes a clock that has the same frequency, e.g. if receiving a 100 kbit/s NRZ data stream the PLL generates a 100 kHz clock. The idea behind exploiting the PLL is therefore straightforward. If the attacker transmits a data at a faster rate the PLL synthesizes a higher frequency data clock, which is the desired outcome. This works as long as the attacker stays within the limits of the PLL, i.e. does not require it to synthesize a clock outside its operational parameters. Speeding up the data clock when the receiver uses the counter method requires slightly more effort but is also possible.

The counter method described requires that the receiver generates a 16× sampling clock, which is then synchronised with the Manchester encoded data. After receiving the first transition, the counter limit is set to 4. Once the counter reaches 4, the receiver samples and sets the counter to 0 and the counter limit to 8. Once the counter reaches 8, the receiver samples and again sets the counter to 0 and the counter limit to 8. It continues in this state until an edge transition occurs, at which time the entire process starts over, i.e. counter limit set to 4 and counter set to 0. By shifting the edge transitions forward the attacker resets the counter early, causing the receiver to sample earlier. As a result the attacker speeds up the sampling process and gains time T_A over the entire data sequence, as shown in Figure 6.9.

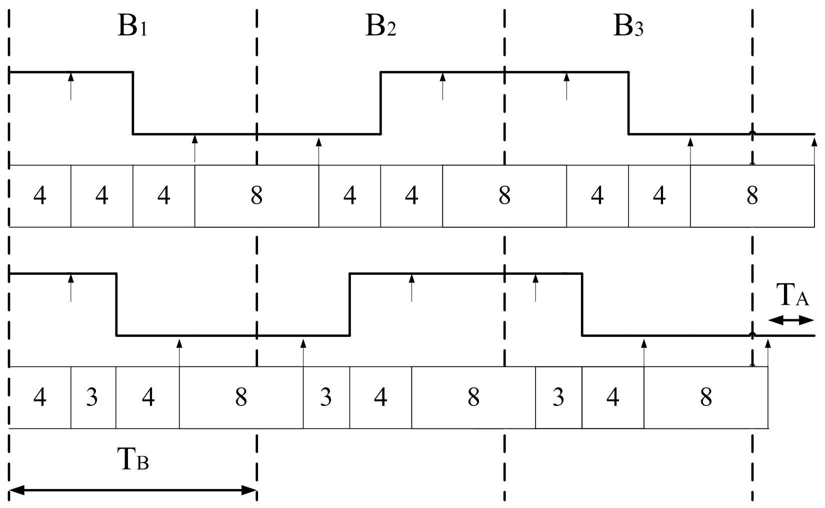


Figure 6.9: Inducing an early response by influencing the receiver's sampling clock. The numbered blocks indicate the value of the counter when it was reset.

6.3 Distance-bounding channels

Conventional communication channels are designed for reliable data transfer. As a result, these channels feature redundancy and timing tolerances to prevent bit errors, which also introduces latency for an attacker to exploit. Systems planning to use distance-bounding protocols must, therefore, implement special low-latency channels. Published proposals for the implementation of distance-bounding channels are currently confined to the HF RFID environment [64,115,137], although there are also some proposals for creating unforgeable RF channels that could prevent relay attacks [69,98,131].

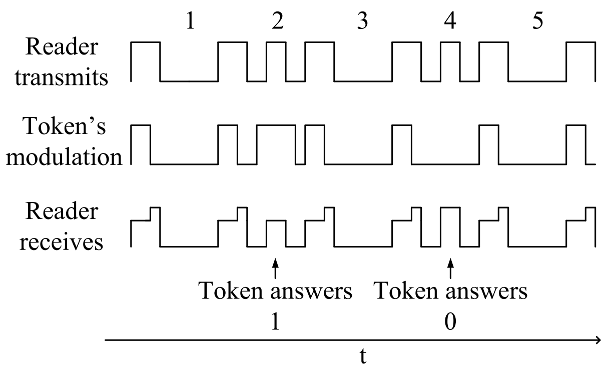
6.3.1 Unforgeable channels

Several proposals attempt to construct unforgeable channels. If the verifier could reliably determine whether the source of the transmission is the prover then relay attacks can be detected. The basic principle in this case is that the proxy-prover will not be able to impersonate the real prover if he cannot exactly replicate the communication channel, e.g. Alkassar, et al. suggested that channel-hopping radio is difficult to track and thus difficult to relay [2]. The verifier can also try to uniquely identify the prover by using the physical characteristics of the channel. Rasmussen and Čapkun proposed that a verifier can construct a unique 'fingerprint' for each prover by using the attributes of the received RF signal [131]. A proposal by DeJean and Kirovski would allow a prover to identify itself by intentionally making its channel characteristics unique [41]. This is achieved by placing a random constellation of conductive and/or dielectric objects within the token, which would alter the near-field response of a token when exposed to RF signals. Neither of these methods provides any accurate proximity information apart from 'in communication range', nor do they protect against a fraudulent prover.

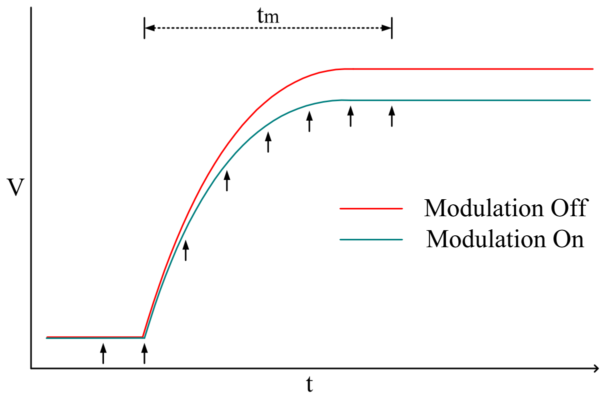
Further proposals hide additional information within the transmitted data. In a scheme by Hu, et al. [69], geographical information, referred to as packet leashes, are added to transmitted data. This method, however, requires the verifier to know its location, which disqualifies it for two-party distance-bounding as it requires collaboration with additional parties. Kuhn [98] proposed that the prover transmits a hidden 'watermark' along with the data, which is subsequently revealed, so that the verifier can retroactively check whether the data it received was transmitted by the prover. This method was suggested for GPS where the sender is trusted, and therefore it does not protect against a fraudulent prover.

6.3.2 Carrier sampling

There are two proposals for a distance-bounding channel where the verifier directly samples the modulated carrier. This means that the verifier could determine the prover's response without performing traditional demodulation and decoding, thus reducing communication channel latency. Both proposals are tailored to the HF RFID environment and depend on the load modulation process, which allows the token to amplitude modulate the carrier transmitted by the reader.



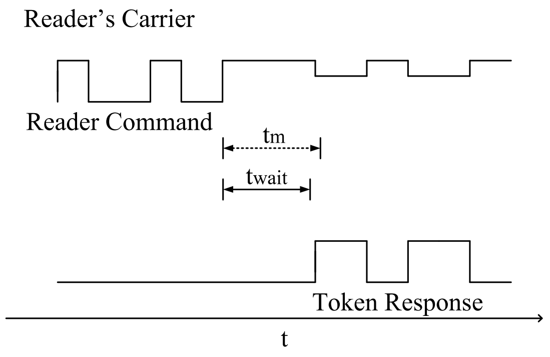
(a) Example of a bit exchange sequence.



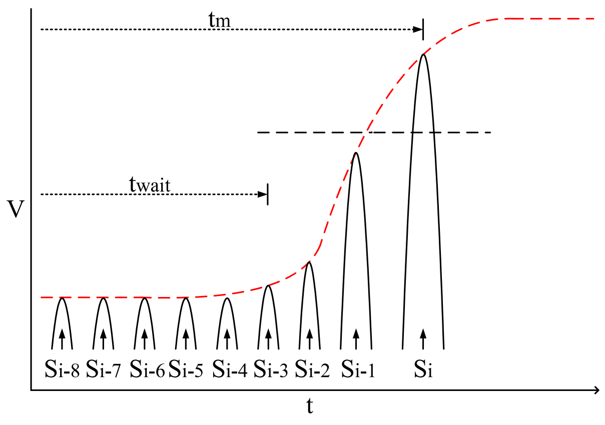
(b) Timing of a single bit exchange.

Figure 6.10: The void-challenge distance-bounding channel.

In the proposal by Munilla, et al. [115], the reader transmits a periodic sequence of pulses that are 100% 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. An example of how successive bits are exchanged is shown in Figure 6.10(a). 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. To determine the tokens's response, the reader continuously samples the envelope of the carrier until it finishes rising and becomes stable, e.g. two successive samples are equal. Once the envelope reaches this steady state the verifier checks the amplitude level to see whether load modulation is on or off. The time it takes until the two levels can be reliably distinguished, and the difference between the envelope amplitude for the two states, depend on the distance between the token and the reader. The reader times from the point when it switches on the carrier, and the envelope starts rising, until the token's response is determined, as shown in Figure 6.10(b). The authors state that the timing resolution of the channel is less than 1 µs. Since the token knows when the reader will issue a challenge, and is in fact expected to respond preemptively, this implementation does not allow for the prevention of distance fraud. The token would also 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) Example of a challenge-response sequence.

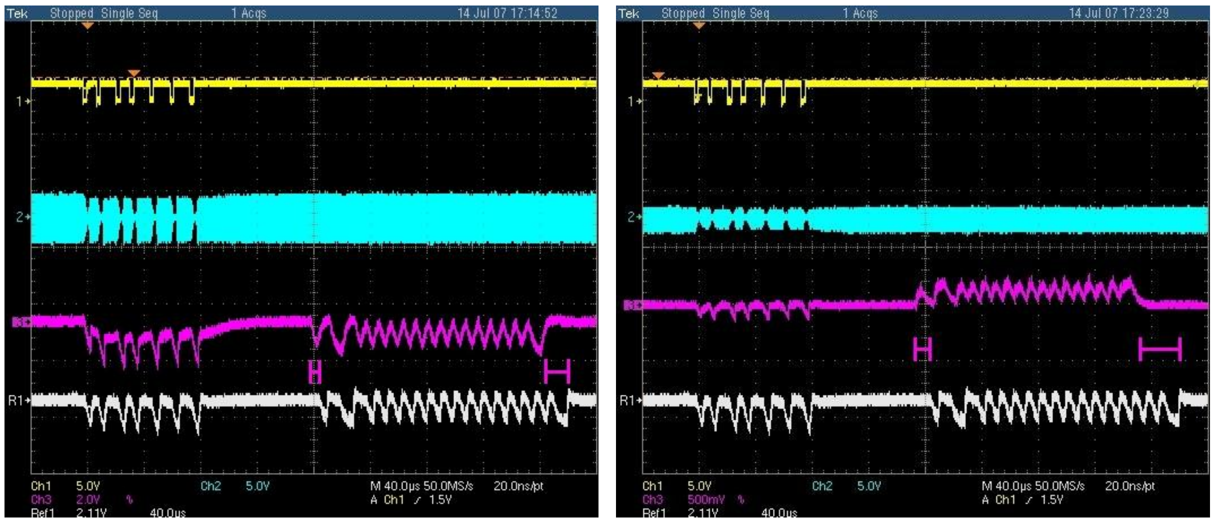


(b) Timing the start of the token's response.

Figure 6.11: Accurately timing the token's response by early modulation detection.

The proposal by Reid, et al. assumes that the token will reply after a fixed time t_{wait} [137]. 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 times from the end of its command to the moment that the response is detected, with the distance-bounding time measurement then taken as $t_{\text{m}} - t_{\text{wait}}$. An example of a challenge-response sequence is shown in Figure 6.11(a). The time at which the response is received is measured using a special detector that tries to determine the exact moment that the amplitude of the carrier is first modulated. This involves 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.

This channel could be vulnerable to distance fraud if the prover does not wait t_{wait} and transmits its response pre-emptively. The authors also state their assumption that the token is protected against overclocking and that the RF carrier operates within the ± 7 kHz tolerance specified by the relevant standard. However, this does not seem to be a valid assumption for tokens currently available. Figure 6.12 shows the effect on the token's response if the frequency of the HF carrier is increased. The first trace shows the *REQA* request sent by the reader, with the second trace showing the corresponding carrier modulation. The third trace shows the token's *ATQA* response, the sequence on the right of the picture, when the carrier frequency is increased. The fourth trace shows a reference token's response when the carrier is 13.56 MHz. In each case the response was recovered using a tuned pick-up coil held close to the token. In Figure 6.12(b) the recovered data is slightly distorted as the operating frequency moves away from the coil's tuned frequency. Accelerating the carrier increases the transmitted bit rate so the final bit is time shifted much more when compared to the time shift of the first bit.



(a) +1 MHz: First edge ≈5 μs, last edge ≈15 μs (b) +2 MHz: First edge ≈10 μs, last edge ≈ 30 μs

Figure 6.12: Time gained from 'overclocking' a 13.56 MHz contactless token

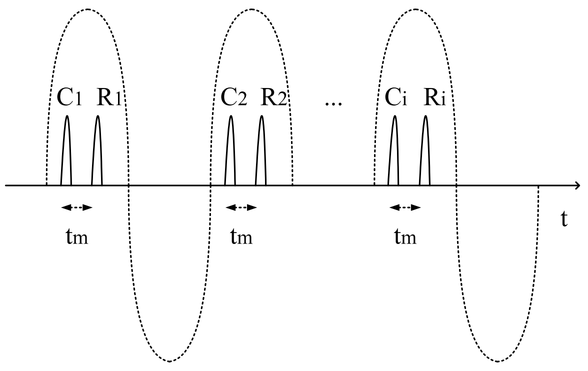
6.3.3 Ultra-wideband pulses

Kuhn and I proposed a crude ultra-wideband channel for near-field systems [64] that adheres to the principles of secure distance bounding defined in Chapter 5. Making the bit period as short as possible would limit the attacks described in this chapter, although this requirement might compromise the reliability of the channel. If this is the case the distance-bounding protocol would need to allow for bit errors during the timed exchange stage. The reader and the token use the 13.56 MHz carrier for loose synchronization and the response is sent immediately after receiving the challenge using an asynchronous circuit which limits the effect of overclocking attacks. A challenge and response bit exchange occurs on each rising edge of the carrier, as shown in Figure 6.13(a).

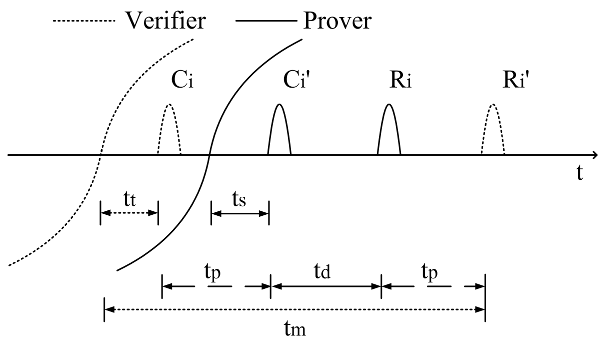
The reader starts timing on the zero-crossing of the carrier, waits for t_{t}, and then transmits the challenge bit C_i. The token also waits for the zero-crossing of the carrier before it starts the sampling process. The sampling time t_{s} is fixed and dependent on the token's hardware implementation. 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'.

In a very simple implementation, the reader is equipped with two adjustable delay circuits. The first delay t_t is used to position the challenge bit such that the token has the best chance to sample the challenge bit correctly. The second delay element t_m is used to time the moment after the zero-crossing when the reader has the best chance to sample the incoming R_i bit correctly. It is up to the reader to repeat the protocol and try different values for t_t and t_m until R_i' matches the expected result well. The total number of bits exchanged n should be chosen large enough such that enough bits remain to satisfy the security requirement of the challenge-response phase after the delay-element adjustment phase. In a more sophisticated implementation the verifier samples a response for multiple delays that are of interest, and then searches in the recorded results for the lowest value t_m with an acceptable response. In neither case are high clock-frequency circuits, or precise reference frequencies, needed in the token. The timing of a single bit exchange is shown in Figure 6.13(b).

The propagation time t_p can then be calculated by the reader as follows: $t_\mathrm{p} = (t_\mathrm{m} - t_\mathrm{t} - t_\mathrm{d})/2$. As with all the other channels presented here, a prover could commit distance fraud if it managed to decrease the expected t_d. Minimizing t_d limits the amount of time the attacker could gain. Drimer and Murdoch recently used a similar technique to implement distance bounding to prevent relay attacks against contact smart cards [45]. In the next section, I describe the practical requirements of our proposal in more detail and show how the channel might be implemented in a near-field device.



(a) Example of a bit-exchange sequence.



(b) Timing of a single bit exchange.

Figure 6.13: Ultra-wideband pulse exchange using carrier synchronisation.

6.4 A near-field bit-exchange channel

In this chapter, the communication channels used in HF RFID systems were shown to be vulnerable to late-commit and clocking attacks. These channels also use traditional communication formats, including error-correction and packet delimiters, and are thus not suitable for distance bounding. An additional communication channel, which shortens the bit period and allows for single bit exchanges would therefore be required to obtain a useful distance bound. Of the current proposals, discussed in Section 6.3, the elementary wideband-pulse channel described in Section 6.3.3 appears to be the most suitable for distance-bounding applications in terms of both timing accuracy and limiting relay attacks and distance fraud. Ultra-Wideband (UWB) communication has already been successfully implemented in active RFID systems for localisation [162], and also proposed for passive UHF RFID systems [121, 174]. UWB communication implements carrierless data transmission. This simplifies the transmitter and receiver architectures, which results in small, inexpensive and low-power hardware. The allocated bandwidth allows for a large channel capacity, while the communication can be made resilient to noise and multi-path effects [57]. UWB channels can also be used for distance measurements with resolution of 30 cm or less [56]. Existing UWB channels are, however, primarily intended for data transfer, which means that they implement packet formatting to synchronise communication and increase reliability, which is not ideal for secure distance bounding applications. In this section, I show how a crude wideband pulse channel for distance-bounding could be practically implemented in the near-field environment, while still allowing for a resource-constrained prover. The implementation permits multiple exchanges of single challenge and response bits, and is tailored to the protocol proposed in Section 5.5.

6.4.1 System considerations

A distance-bounding channel provides the verifier with a timed measurement, from which to determine the distance bound. This measurement must be as accurate as possible and should not include any additional latency that can compromise the security of the protocol. A practical implementation, however, must also take into consideration the limitations of the hardware and the operating environment. Some key issues influencing the design are:

- **Hardware constraints:** I assume that the verifier, e.g. the RFID reader, has more resources compared to the prover, e.g. the RFID token. Complex operations, such as variable delay lines and the generation of high-frequency sampling and synchronization clocks, should therefore be implemented by the verifier. For a reliable distance bound the prover's hardware must have a predictable processing delay t_d. The prover's system clock is derived from the received HF carrier, which means that it is not trusted. As a result, the prover should implement asynchronous circuitry that functions independently of the system clock.

- **Synchronisation:** The prover and verifier require a synchronization signal to exchange pulses and provide a timing reference. The channel proposal in Section 6.3.3 suggests that the prover and verifier trigger their operations on a zero-crossing of the carrier transmitted by the verifier. Using the carrier for loose synchronization is a possibility, although it might not be feasible for a bit exchange to occur on every zero crossing, i.e. this would require a low-resource 13.56 MHz token to transmit a bit every 36 ns. The prover and the verifier could, however, use the carrier to generate a lower-frequency synchronization signal. Other possibilities are the transmission of a preceding timing pulse, followed by the data pulse, or switching the carrier on and off.

- **Timing and distance resolution:** The timing accuracy influences the resolution of the distance estimate. The simplest timing method is to start on the synchronization signal, wait for a fixed time t_m, and then sample once. If the correct response is sampled, the verifier can calculate the propagation time t_p from the round-trip time

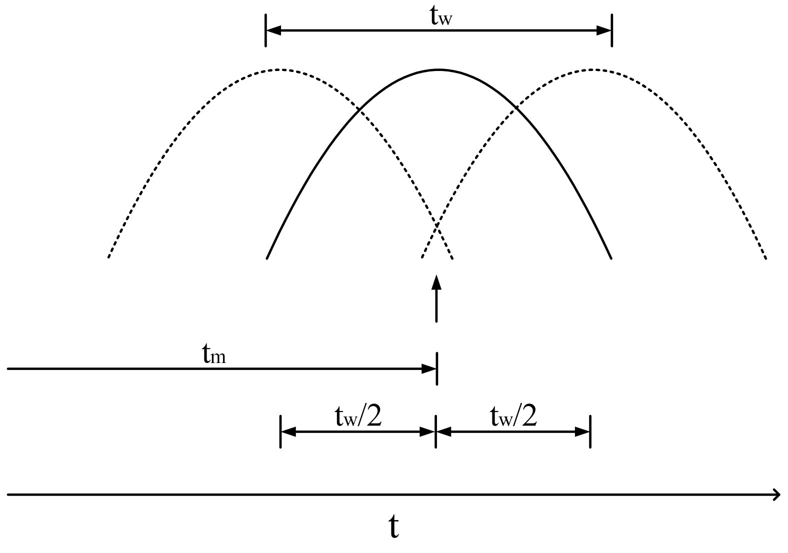


Figure 6.14: The effect of sample timing and pulse width on the distance bound.

$t_\mathrm{m} = 2 \cdot t_\mathrm{p} + t_\mathrm{d}$ and estimate the distance to the prover as $c \cdot (t_\mathrm{m} - t_\mathrm{d})/2$. It should be noted that this estimate only bounds the distance to the prover as the width of the response pulse t_w introduces some uncertainty. As shown in Figure 6.14, the actual distance to the prover can be anywhere between $c \cdot (t_\mathrm{p} - t_\mathrm{w}/4)$ and $c \cdot (t_\mathrm{p} + t_\mathrm{w}/4)$. This situation arises because the verifier does not know the exact time the response arrived, only that it sampled during the response. The verifier could have sampled right at the end, or the beginning, of the response, which means that t_m measured is $t_\mathrm{w}/2$ greater, or less, than the actual round-trip time. The resolution could be improved by sampling multiple times or decreasing t_w.

- **Bit representation:** The transmitted symbols that indicate the value of the challenge, or response bit, also affect the time measurement. In the ideal case, the symbols will be chosen in such a way that the verifier can distinguish between the different symbols, and also detect when no response is received. For example, an on-off keying scheme would make distance estimation more complicated since the verifier does not know whether it received a '0' or whether the response arrived before, or after, he sampled the channel. The two different symbols should also provide identical timing information. Symbols based on pulse-position schemes are therefore not suitable as these provide ambiguous timing, i.e. is the pulse in the second time slot of the symbol, or is the pulse in the first time slot of a symbol transmitted later? Even if synchronization prevented this ambiguity, an attacker immediately knows that he needs to transmit a pulse in the second slot if the first is empty. If the attacker could relay this finding before the second slot started he would provide the correct response, within the required time, in approximately half of the exchanges.

- **Security of the distance bound:** I assume that a relay attacker will not cause any additional delay apart from the time it takes for the signal to propagate over the extra distance, and that he cannot decrease the processing time of the prover. In theory, the prover's asynchronous logic and delay lines could be influenced, e.g. changing the temperature, although this would be difficult without having physical access to the prover's token, for example when covertly reading an RFID token in the victim's pocket. This scenario is more applicable in the case of distance fraud. I assume that a dishonest prover can decrease t_{d} to zero, although the signal propagating over a longer distance will cause additional delay.

6.4.2 Design

The channel described in Section 6.3.3 is, in my opinion, the most suitable for implementing secure distance-bounding for RFID tokens when taking into account the principles discussed in Section 5.4.5. The initial proposal did include some ideas on how to implement the channel, e.g. variable delay line to aid in synchronization and shift registers for storing R^0 and R^1, but practical issues such as generating synchronisation from the shared carrier, defining an accurate system timing model and implementing suitable pulse transmission, lookup and delay line circuits were not addressed. Subsequently, I proposed a more detailed practical implementation for this channel, as shown in Figure 6.15. The symbol definitions and corresponding timing diagram are shown in Figures 6.16 and 6.17, while relevant variable definitions are shown in Table 6.1.

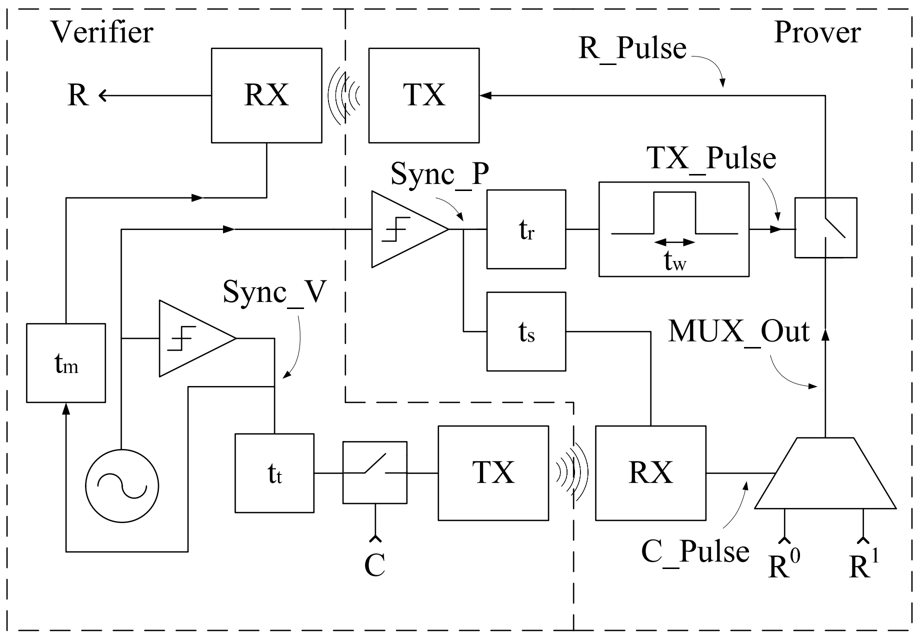


Figure 6.15: Overview of a wideband pulse distance-bounding system.

The verifier starts by generating a suitable synchronization signal, e.g. an RF carrier. This signal is used to generate a clear timing reference, such as a rising edge, by both the verifier (*Sync_V*), and the prover (*Sync_P*). If the challenge bit C is '1' the transmitter will wait t_{t} after the timing reference and then instruct the transmitter to send a pulse. The time taken by the transmitter to generate and drive the pulse signal onto the antenna is t_{TX}. The transmitter's output is represented in the timing diagram by the signal V_{TX}. The prover waits for t_{s} after the timing reference signal and then samples its receiver's output P_{RX} to produce the signal *C_Pulse*. The prover allows for the time t_{RX} its receiver takes to detect and amplify the pulse signal. *C_Pulse* is then used to switch a multiplexer to select one of R^0 or R^1 as a response. The output of the multiplexer *MUX_Out* cannot be connected directly to the prover's transmitter. Due to the physical characteristics of a multiplexer, i.e. one path requires an additional NOT function, the time it takes to change the output when the input changes from '1' to '0' might be different from when the input

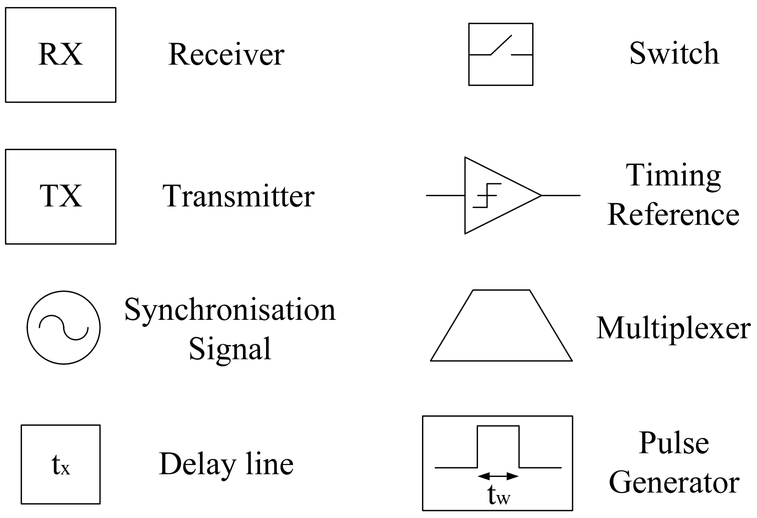


Figure 6.16: Symbol definitions.

changes from ‘0’ to ‘1’. This would make t_{d} dependent on C and complicate the round-trip timing. Connecting the multiplexer output directly to the transmitter also introduces a security vulnerability since it is possible for both R^0 and R^1 to appear on the multiplexer's output during each exchange cycle of the protocol. For example, if the previous *C_Pulse* was ‘1’ then the multiplexer will output R^1 to start with, and then changes to R^0 if the current challenge is ‘0’. This might allow a proxy-prover to read out both response sequences, which would allow a proxy-prover to execute the protocol with 100% success. From a practical perspective the transmitter also requires a rising edge to generate a response pulse, which would not happen if two consecutive responses are ‘1’. These issues are solved by adding a switch to output the current response to the transmitter once the output of the multiplexer is stable. The prover waits for time t_{r} before generating a pulse *TX_Pulse* which switches the multiplexer output through to the receiver. The ‘off’ state of the switch is ‘0’ so a reponse of ‘1’ will also generate the required rising edge to drive the transmitter. The falling-edge of *TX_Pulse* can also be used to clock in the next values of R^0 and R^1. The width of the resultant *R_Pulse* can be adjusted by changing the width t_{w} of *TX_Pulse*. The response is transmitted back to the verifier, once again incurring some transmitting, receiving and propagation delay in the prover's transmitter and verifier's receiver, as indicated by signals P_{TX} and V_{RX} . Finally, the verifier samples its receiver's output t_{m} after the timing reference to determine response R.

The delay lines in the prover, t_{r} and t_{s}, are fixed and it is assumed that these, along with t_{d}, are predictable and known by the verifier. The delays in the verifier, t_{s} and t_{m}, are adjustable and should be configured during the early stages of the protocol. The expected round-trip time is $t_{\mathrm{m}} \approx 2 \cdot t_{\mathrm{p}} + t_{\mathrm{d}} + t_{\mathrm{e}} + t_{\mathrm{RX}}$, where t_{e} is the synchronization error between the time references in the prover and verifier. In near-field communication systems the expected propagation time is almost neglible, i.e. 10 cm is 300 ps, and the verifier should have an estimate for $\mathrm{t_d}$ and t_{RX}, so it can make a reasonable first approximation of t_{m}. At this stage the prover keeps responding with a ‘1’, while the verifier increases t_{m} to the minimum value at which this response can be sampled reliably. Once the verifier completes his adjustment it transmits a nonce to the prover, which indicates that the first calibration step is complete, and is also used to calculate R^1 and R^0. During the initial cryptographic exchanges the verifier will adjust t_{t} to ensure that the prover samples C reliably. It is assumed that the prover cannot provide the correct response if it does not

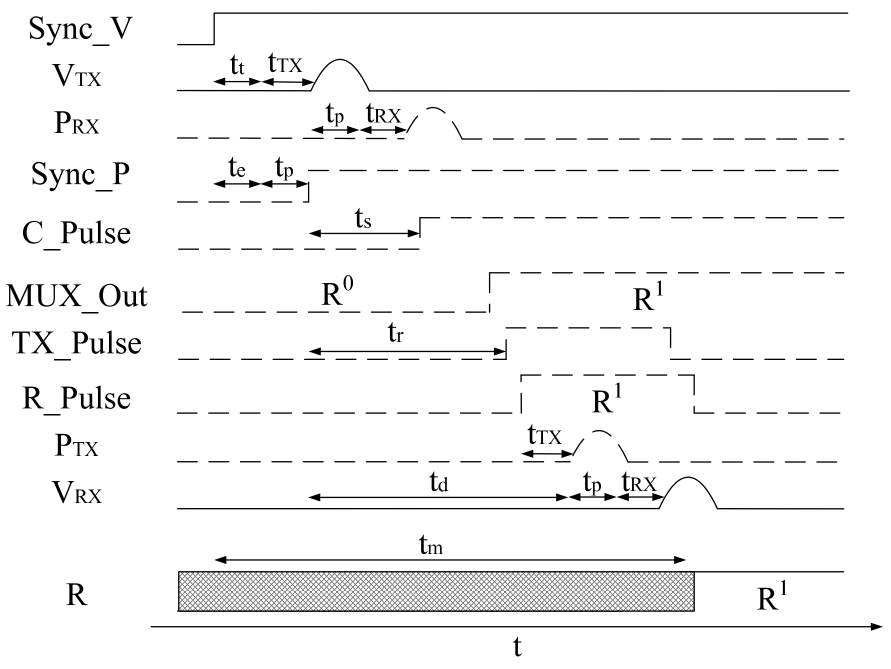


Figure 6.17: System timing diagram.

t_m	Round Trip Time measurement taken by Verifier
Sync_V	Synchronisation signal generated by Verifier
Sync_P	Synchronisation signal received by Prover
t_t	Time delay after *Sync_V* that the Verifier transmits challenge C
t_s	Time delay after *Sync_P* that the Prover samples challenge C
t_e	Synchronisation time error between *Sync_V* and *Sync_P*
t_p	Propagation time between Verifier and Prover
t_r	The Prover's response delay, i.e the time between *Sync_P* and calculating the response
t_d	The Prover's processing time, i.e. the time between *Sync_P* and transmitting response R
t_w	The width of the challenge/response pulse
t_TX	Time delay in transmitter circuit of prover and verifier
t_RX	Time delay in receiver circuit of prover and verifier

Table 6.1: Variable definitions

receive the correct C. The verifier sets t_t to the maximum value at which the majority of the responses it receives are correct, which should results in $t_\mathrm{t} \approx t_\mathrm{e}$. t_s should have been set to $t_\mathrm{TX} + t_\mathrm{RX} + t_\mathrm{p}$, with $t_\mathrm{p} \to 0$, so if the prover cannot sample the challenge this is due to an error in the synchronization of the timing references. The uncertainty introduced by this synchronisation error can be accommodated in the distance bound as it is effectively measured by the value of t_t. Once t_m and t_t have been set, the verifier can evaluate the distance bound. Ideally, the verifier wants to calculate $d = c \cdot t_\mathrm{p}$ but he only has an approximate round-trip time measurement t_m, which includes various processing delays in addition to $2 \cdot t_\mathrm{p}$. It should also be kept in mind that the timing resolution of t_m is dependent on the pulse width and sampling method used, as described in Section 6.4.1. If the response is sampled once, as shown in Figure 6.14, the actual round-trip can be anywhere between $t_\mathrm{m} - t_\mathrm{w}/2$ and $t_\mathrm{m} + t_\mathrm{w}/2$. As the *upper* bound, i.e. the furthest distance the prover can be from the verifier, is required the verifier should allow for the latter time in his distance bound calculation. From Figure 6.17 and taking into account that $t_\mathrm{t} \approx t_\mathrm{e}$ the round-trip time can be defined as follows:

$$t_\mathrm{m} \approx 2 \cdot t_\mathrm{p} + t_\mathrm{d} + t_\mathrm{t} + t_\mathrm{RX} + t_\mathrm{w}/2 \tag{6.1}$$

When the distance bound is calculated two attacks should be considered: a relay attack by a third party attacker and distance fraud by a dishonest prover. For the relay attack I assume that the third party attacker, who controls a proxy-prover and proxy-verifier, introduces no extra delay except the additional propagation time of the relayed communication. A third party attacker cannot decrease delay times, such as t_r and t_s, in the real prover as these are determined by asynchronous delay lines. There are arguments that delay lines can be sped up by cooling the token [45], but unless the prover's hardware is controlled by the attacker, in which case it should be seen as a distance fraud, it is not practically feasible to remotely cool the circuit, e.g. it is difficult to covertly spray liquid nitrogen on a victim's purse or pocket. As a result, the attacker cannot make the real prover answer earlier than expected and the verifier can store an accurate estimate of the processing time t_d of the real prover. Similarly, I also assume that the attacker cannot manipulate the delay introduced by the receiver circuits in the prover and verifier. As a result the verifier also has an accurate estimate of the delay t_RX caused by his receiver circuitry. The verifier can then subtract the estimates of t_d and t_RX along with t_t, which was set earlier by the verifier, from t_m to get an accurate approximation of t_p. In the case of a relay attack the distance bound d can therefore be calculated as follows:

$$d = c \cdot (t_\mathrm{m} - t_\mathrm{t} - t_\mathrm{d} - t_\mathrm{RX})/2 \tag{6.2}$$

Substituting the approximation for t_m given in Equation 6.1 this simplifies to

$$d = c \cdot (t_\mathrm{p} + t_\mathrm{w}/4) \tag{6.3}$$

A dishonest prover can decrease his processing time t_d by implementing new receiver, transmitter and response look-up circuits that introduce less delay. Although it is probably not practically feasible, I consider the worst case scenario and assume that the dishonest prover could reduce his processing time t_d to zero, i.e. reply instantaneously without any delay. If $t_\mathrm{d} = 0$ is substituted in Equation 6.1 the round-trip time measurement taken by the verifier becomes

$$t_\mathrm{m} \approx 2 \cdot t_\mathrm{p} + t_\mathrm{t} + t_\mathrm{RX} + t_\mathrm{w}/2 \tag{6.4}$$

Substituting this modified estimate of t_m into Equation 6.2 results in

$$d = c \cdot ((t_\mathrm{p} + t_\mathrm{w}/4) - t_\mathrm{d}/2) \tag{6.5}$$

If the distance bound is calculated in this way, a fraudulent prover could therefore be up to $d = c \cdot t_\mathrm{d}/2$ further away and still appear within acceptable distance. For example, if the processing time is decreased by 12 ns then the dishonest prover could afford an additional 6 ns in the propagation time t_p, which means he can be approximately 2 m further from the verifier without his fraud being detected. A verifier allowing for distance fraud should therefore take into account that his expected value for t_d is not correct. As a result, the verifier does not use this estimate when calculating the distance bound. This approach results in a valid distance bound for both honest and dishonest provers, even though the actual distance to a honest prover is probably less since he does introduce processing delay t_d. It should, however, be remembered that the distance bound is not the actual distance to the prover. It is simply an upper bound on the distance between the verifier and the prover, i.e. the prover is within d, and if allowing for a dishonest prover the worst case should be taken into account. If the verifier allows for possible distance fraud then the distance bound d should be calculated as

$$d = c \cdot (t_\mathrm{m} - t_\mathrm{t} - t_\mathrm{RX})/2 \tag{6.6}$$

Substituting the approximation for t_m given in Equation 6.1 this simplifies to:

$$d = c \cdot (t_\mathrm{p} + t_\mathrm{d}/2 + t_\mathrm{w}/4) \tag{6.7}$$

6.4.3 Practical implementation

I performed some practical experiments focusing on three key functions: synchronisation, pulse transmission and asynchronous lookup. The electronic circuits described here are by no means the only solution, nor are they meant as reference designs for a commercial distance-bounding channel. My goal was only to show that it is possible to implement these functions within a near-field environment using limited resources. The basic building blocks, e.g. delay lines, are based on reference designs, which I obtained from electronic circuit sources like [68].

Synchronisation

Earlier it was suggested that the carrier could be used for providing a time reference point. This was, however, not as straight forward as it appeared at first. The transmitted and the received versions of carrier are not exactly synchronised. This is to be expected since the prover token is in effect a complex load, which will cause a phase shift if the token and the verifier's reader are not impedance matched. I have observed the output of several tuned resonant circuits and in general the error was minimal ($t_\mathrm{e} < 10$ ns), with some exceptions ($t_\mathrm{e} \approx 25$ ns). The synchronisation error should, however, compensated for in the distance bound calculation by the value of t_t.

If the unmodified carrier is used for synchronisation, the period of the HF carrier places a restriction on the maximum round trip time that can be measured. In a 13.56 MHz RFID system, t_m would need to be less than 73 ns since the prover should ideally be allowed to respond before being issued with another challenge. This also places restrictions on t_d, t_RX and t_t, which complicates the channel further. The prover and the verifier should therefore use a lower frequency synchronization signal derived from the carrier. This could be done by implementing a frequency divider, i.e. binary counter, in both the prover and verifier. At the start of the protocol the counters would need to be synchronised. This could possibly be done by resetting them if the carrier is switched off. Once the carrier is switched on again, both counters start on the first rising edge of the carrier. Their output will obviously also be influenced by any carrier synchronisation error.

Transmitting a pulse while the carrier is on also complicates the receiver architecture, as the receiver needs to distinguish a relatively weak pulse in the presence of a strong carrier. An alternative solution would be to switch the carrier off and use an envelope detector in both the prover and the verifier to perform synchronisation. When the verifier wishes to transmit a challenge it switches off the carrier and both parties generate a time reference when the falling edge of the envelope passes through a set threshold. The difference in the amplitudes of the transmitted and the received carrier can be compensated for by making the threshold a percentage of the average carrier amplitude, rather than a fixed reference value. The synchronisation accuracy is effected by the component tolerances and phase of the carrier in the prover and verifier. Once again, these factors are compensated for in the distance bound calculation by the value of t_t.

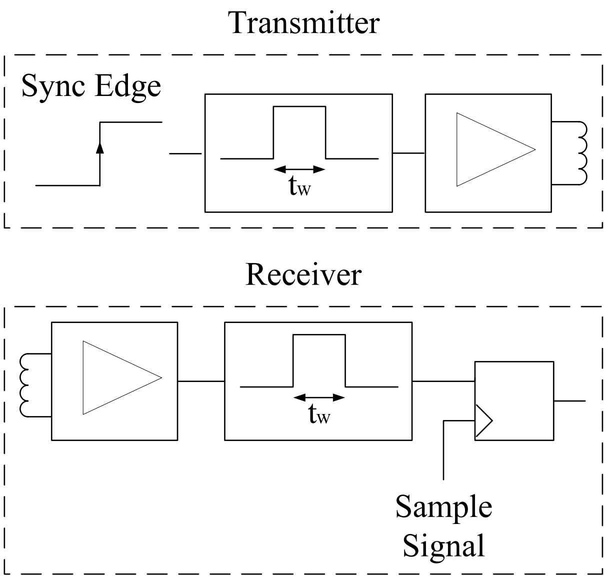


Figure 6.18: The transmitter and receiver architectures.

Pulse transmission

The simplest solution is to implement an on-off keyed pulse channel, i.e. pulse is ‘1’, nothing is a ‘0’. The only drawback of this channel is that there will be no verifiable timing information when the response is ‘0’, as nothing will be transmitted. This would make this channel unsuitable for doing distance estimation using a single exchange. However, taking into account that the channel is meant to be used in a distance-bounding protocol performing multiple exchanges, and that the verifier is only interested in the answer at time t_m, this method is sufficient. The transmitter and receiver architectures are shown in Figure 6.18.

The transmitter generates a pulse of width t_w when a rising edge occurs on its input. The pulse generator can be constructed in a similar way as described earlier. The pulse is then transmitted using a buffer capable of sourcing enough current to drive the signal onto a small loop antenna. This buffer could be an amplifier, or simply multiple logic gates with the same output. The receiver also uses a small loop antenna, connected to an amplifier. This architecture works well when the pulse is not transmitted at the same time as the carrier. The first rising edge of the received signal is then used to generate a ‘new’ pulse of width t_w. This provides the sampler with a clean input pulse, irrespective of the quality of the received signal. The sampler is a D-type flip-flop (74HC74), clocked by an external sampling signal.

Asynchronous response circuit

The prover needs to implement two delay lines, a multiplexer and a response switch. I assume that R^0 and R^1 are pre-loaded into two shift registers, and that they are both clocked onto the multiplexer inputs well before C is received. Alternatively, the next (R^0, R^1) pair can also be computed by iterating a pseudo random-bit generator well before C is received. A circuit that could be used to choose, and clock out the required response is shown in Figure 6.19. The entire circuit can be implemented using discrete logic and passive components.

The multiplexer is implemented using NOT (74HC04) and NAND (74HC00) logic gates, and outputs R^x where x is the value of the input. The input, *C_Pulse*, is the value sampled by the receiver after delay t_s. A delay line is implemented using NOT logic gates and an RC-network, which decrease the rise, or fall, time of the applied edge. Since the signal rises, or falls, more slowly it reaches the $L \leftrightarrow H$ threshold of the next gate later, which results in the output being a delayed version of the original signal edge. A similar delay line is implemented for t_r. The delayed edge then triggers a pulse generator, built using NOT gates and an RC-network. If a rising edge is applied to the RC-network the output immediately goes to a high level, but as the capacitor charges the level drops down again. If this charge-discharge cycle is applied to the input of a logic gate a pulse is generated at the output. In this case the positive pulse *TX_Pulse* is used to switch the output of the multiplexer to the transmitter. The switch is implemented using an AND (74HC08) logic gate.

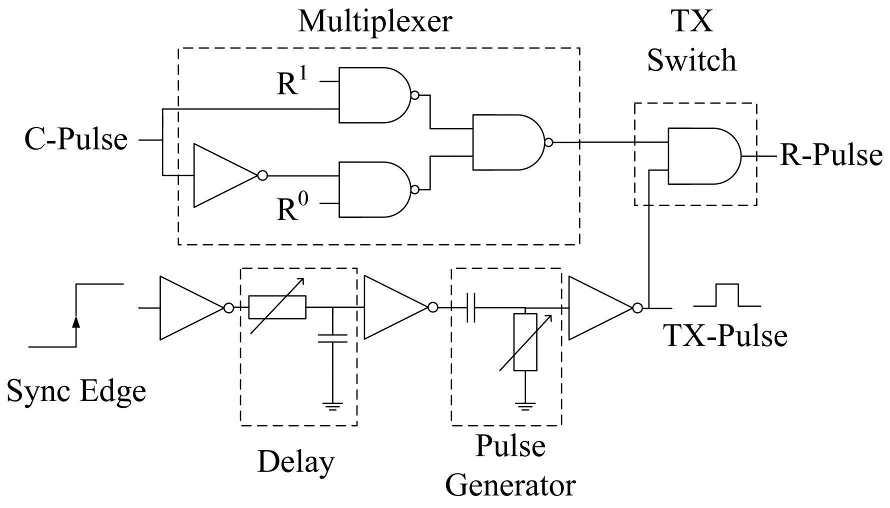


Figure 6.19: The prover's asynchronous circuit for choosing the response.

Experimental results

I implemented a simplified version of the proposed distance-bounding channel design. The experimental hardware implementation included:

- Two TX–RX links implementing a duplex channel between the prover and verifier.
- Delay lines t_t, t_m, t_r and t_s.
- An asynchronous lookup circuit.

C, R^0 and R^1 was implemented with user controlled switches and the delay lines t_{m} and t_{t} was adjusted manually. For the experiment a 100 kHz clock signal, connected to both the prover and verifier, was used for synchronisation. This is not realistic, but allowed me to simulate synchronization errors by delaying the signal to the prover. Some initial results are shown in Figure 6.20. The width of the transmitted pulses was approximately 10 ns, although in the figure the prover's receiver generates a wider *C_Pulse*. This was to show that a narrow pulse can be stretched to provide the prover with a greater time window to sample correctly. This would simplify the initial setup when adjusting t_{t} as it shortens the time taken before the prover samples at the right time and starts responding with the correct responses. The security of the distance bound is not affected since t_{t} will still be adjusted to the maximum value, i.e. where the prover samples just after the pulse edge. If t_{e} was zero the experimental system had a round trip time t_{m} of approximately 75 ns. t_{RX} of the verifier was approximately 14 ns.

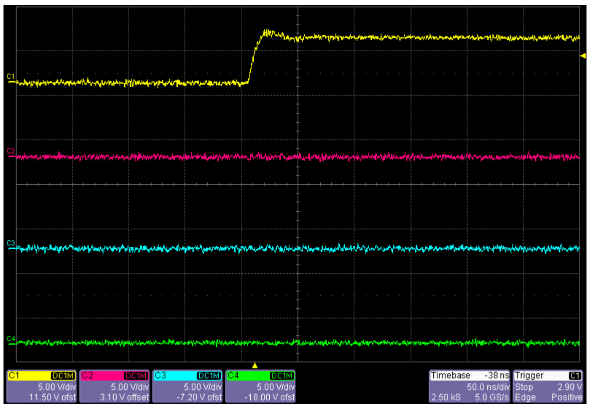
Given that the propagation time in a near-field system is negligible, $t_{\mathrm{p}} \approx 300$ ps while $t_{\mathrm{m}} \approx 75$ ns, t_{p} effectively tends toward zero. As a result Equation 6.3 on page 138 and Equation 6.7 on 139 can be modified to $c \cdot (t_{\mathrm{w}}/4)$ and $c \cdot (t_{\mathrm{d}}/2 + t_{\mathrm{w}}/4)$ respectively. If the measured round-trip time t_{m} in this channel implementation is 75 ns the upper bound on the distance is calculated as follows:

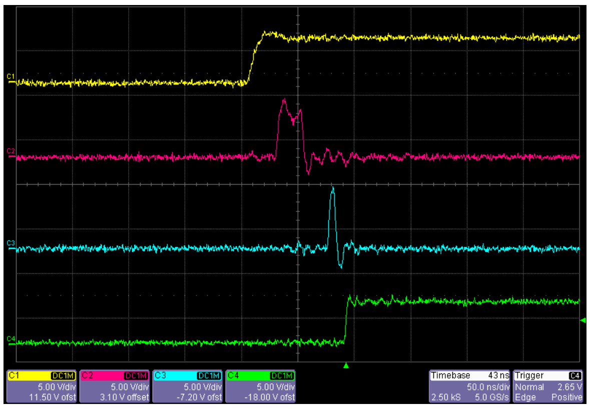
- An honest prover is within $d = c \cdot (2.5 \text{ ns}) \approx 1$ m of the verifier, even if a third party attacker executed a relay attack.
- If the prover is not trusted and in a position to execute distance fraud, the verifier concludes that the prover is within $d = c \cdot (60 \text{ ns}/2 + 2.5) \approx 11$ m.

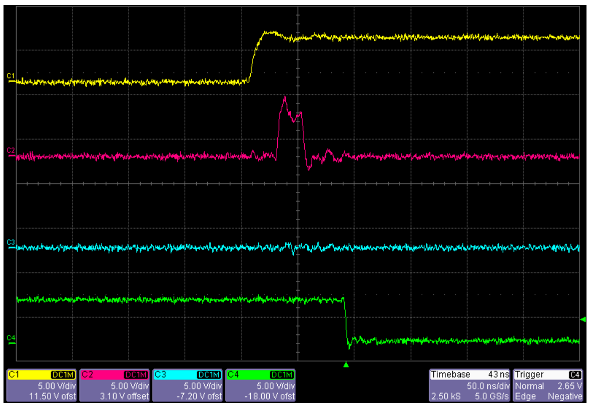
6.5 Conclusion

The security of a distance bound depends not only on the cryptographic protocol used but also on the practical implementation and the physical attributes of the communication channel. The communication channel used for the exchange must not introduce any latency that the attacker can exploit to circumvent the physical distance bound. I show how an attacker can implement 'late-commit' attacks against receiver architectures by exploiting latency introduced during the demodulation stage. In the two receivers I tested, an attacker only needs to commit to a bit value 22 μs and 2.5 μs after the start of the bit period. This would lead to errors of 3.3 km and 375 m, respectively, if these channels were used for time-of-flight distance bounding. I further discuss how an attacker could exploit the decoding step by only committing to a bit value slightly before the decoder samples the signal. The attack time in these cases would be dependent on how often the decoder samples, and whether it uses a majority voting scheme or samples only once. I also show how an attacker can speed up the reply of the prover by influencing the receiver's recovered data clock or, in the case of RFID tokens, the system clock of the prover. For example, a 14443 A token I tested started transmitting an ATQA response 10 μs early and finished the response 30 μs early when the carrier frequency was increased by 2 MHz.

The communication channel vulnerabilities presented in this chapter undermine the security of distance-bounding protocols. This leads me to conclude that conventional receiver architectures are unsuitable for implementing distance-bounding protocols. Special


(a) $C = 0$ and $R = 0$


(b) $C = 1$ and $R = 1$


(c) $C = 1$ and $R = 0$

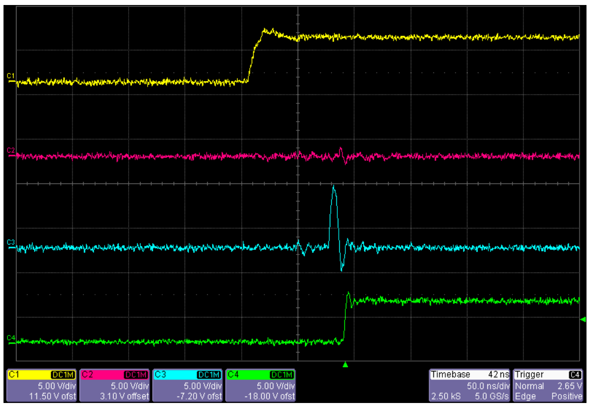

(d) $C = 0$ and $R = 1$

Figure 6.20: Different bit exchanges on the experimental distance-bounding channel. The top trace shows the synchronising edge, the second trace shows the output of the prover's receiver, the third trace shows the output of the verifier's receiver and the bottom trace shows the response sampled by the verifier after t_m. The initial state, the left hand side, of the bottom trace shows the response sampled during the previous exchange.

consideration must, therefore, be given to the communication channel used for distance bounding, and the designer must include any potential vulnerabilities into the final distance bound estimate. Ideally, distance bounding should be implemented using a specially designed channel. Current proposals for relay-resistant channels show some promise but are not yet perfect. For example, unforgeable channels provide no distance information, and might be vulnerable to distance fraud, while both carrier sampling techniques presented for the RFID environment fail to protect against distance fraud.

A channel exchanging single short pulses would offer the best distance-bounding characteristics but would require specially designed hardware. I therefore discuss how a bit-exchange channel could be implemented in the near-field environment using an improvised wideband-pulse channel. I propose a design taking into account the identified system considerations and describe how key functions of this design could be achieved using simple hardware. Some practical results are also presented from an experimental bit-exchange channel based on my design. The experimental hardware bounds $d \approx 1$ m for an honest prover and $d \approx 11$ m for a fraudulent prover. I hope that this work will encourage further research on implementing communication channels for distance-bounding protocols.

Chapter 7

Review and further work

The core aim of my dissertation was to illustrate the weaknesses in current proximity identification systems using RFID devices, and to investigate the possible use of distance-bounding protocols to make these systems more secure. This review briefly highlights research contributions made toward the areas of RFID security and proximity identification. For more detailed results and conclusions the reader should consult the final section of each chapter. The main contributions in this dissertation are:

- **Practical results for eavesdropping and skimming attacks**: These attacks are a well known risk for RFID devices, yet very few publications give details about possible experimental setup or actual results. I successfully implemented practical proof-of-concept eavesdropping and skimming attacks against HF RFID devices. In each case I describe the experimental attack setup in detail. This work provides security researchers with a reference attack and also contributed toward a baseline threat assessment for HF RFID tokens.

- **Eavesdropping resistant channel using cover noise**: I showed that current bit-blocking schemes used to obfuscate RFID data are vulnerable because attackers can distinguish between the blocking sequence and the data. I propose a method where the reader transmits additional AWGN on its carrier to make it more difficult for an attacker to distinguish between the blocking sequence and data on the backward communication channel. I presented simulated results suggesting that this method increases the probability that an attacker will make significant bit-errors when attempting to recover the data. This scheme could be used for key exchange between near-field devices with limited cryptographic resources and also by RFID proxy systems.

- **Practical demonstration of a relay attack against HF RFID systems**: I demonstrated that HF RFID systems are vulnerable to relay attacks by implementing a practical attack against a system adhering to ISO 14443 A. I discussed the security implications of this attack and also provided a detailed description of the experimental attack hardware. The hardware can also be used to implement active relay attacks where the relayed data is modified en-route. I hope that the relative simplicity of this attack, especially in the RFID environment, will increase interest in distance-bounding methods.

- **Analysis of distance-bounding protocols**: I investigated the possibility of providing a cryptographic proof of physical proximity by using distance-bounding protocols. Several methods for distance estimation were discussed and I explained why the ToF measurement of RF signals are most suitable to secure distance bounding. I described different ways of implementing distance-bounding protocols and reviewed a number of published proposals. Several protocol proposals were shown to fail if bit errors occur during the exchange stage. Some proposals also fail to consider the latency, introduced by processing responses and the underlying communication channel, which an attacker can exploit to circumvent the distance bound.

- **General design principles for secure distance-bounding protocols**: I discussed a number of principles to adhere to when implementing distance-bounding systems. These seek to address common weaknesses found in current distance-bounding proposals. The principles restrict the choice of communication medium, the communication format and recommend that protocols be made tolerant to bit errors.

- **A distance-bounding protocol for HF RFID systems**: I described a new distance-bounding protocol that is suitable for resource constrained provers. The prover (token) only needs to implement a pseudo-random function and simple asynchronous look-up circuitry, while the verifier (reader) performs complex functions like clock generation and fine resolution timing. This protocol could therefore be implemented on RFID tokens ranging from simple tags to contactless smart cards. The protocol assumes that the token and reader implement a 'slow' error-corrected channel for data transmission and a 'fast' bit-exchange channel to provide accurate time measurements. The proposed protocol only sends a single nonce from the verifier to the prover during the setup stage and requires no verification stage. This not only simplifies its implementation, but also limits the communication on the 'slow' channel, which makes it suitable for applications where a rapid completion of the protocol is required. The protocol is also resistant to bit-errors occurring during the 'fast' bit exchange.

- **Importance of the physical communication channel in secure distance bounding**: Distance-bounding protocols rely on the physical layer of the communication channel for accurate time measurement used to estimate distance. I showed how an attacker can exploit latency introduced during the demodulation and decoding steps to execute 'late-commit' and overclocking attacks against RF receiver architectures often used in sensor networks and RFID systems. Conventional RF channels are therefore not suitable for implementing secure distance-bounding protocols. Finally, I looked at existing proposals for distance-bounding and relay-resistant communication channels and comment on their effectiveness.

- **A distance-bounding channel for near-field devices**: I discussed how a bit-exchange channel using wideband pulses could be implemented in the near-field environment using an improvised wideband-pulse channel. I proposed a possible design taking into account the identified system considerations and described how key functions of this design could be achieved using simple hardware. Some results from a practical prototype implementation of the proposed bit-exchange channel were also presented. I hope that this work will lead to further research being done on implementing communication channels that will support distance-bounding protocols.

Some observations presented in this dissertation that might warrant further investigation or could lead to future projects are:

- My experimental eavesdropping setup involved data transfer from the oscilloscope to a PC, where the data was further processed with MATLAB programs. In addition, the RF receiver's size and weight make it unsuitable for use in a portable eavesdropping system. I discussed the implementation of a simple attack receiver suitable for a covert attack, albeit with a shorter eavesdropping range than the commercial receiver, but further work could focus on implementing additional hardware that can process the received RF signal and store, or output, the recovered data without the need for additional measurement equipment. Such an RF 'sniffer' could be used to study RFID protocols or as a communication interface debugging tool.

- Eavesdropping results are dependent on environmental noise. Further work can be done to develop a test-bed for studying how man-made environmental noise affects eavesdropping attacks. Practical experiments would require a configurable noise source that could be used to simulate various environments. This test-bed can also be used to investigate whether intentionally adding noise is a practical solution for creating eavesdropping resistant communication channels.

- It has been proposed that existing NFC devices could be used as an attack platform for relay attacks. Even though the deployment of NFC devices are currently limited it would be interesting to investigate whether this would be practically possible once these become available.

- High-volume RFID tokens often have limited security features, yet they are used to store valuable information. I already mention that some tokens appear to have no tamper resistance and that tokens can be overclocked. Further work could investigate whether these tokens are vulnerable to simple hardware attacks, previously described for early generation smart cards, or examine the proprietary ciphers and protocols currently implemented, e.g. the Crypto1 algorithm used by Mifare Classic tokens has recently been reverse engineered [119].

- I described the possible implementation of a distance-bounding channel for near-field devices and implemented an experimental system to illustrate the main functionality. Further work could be done on optimising this design and constructing a more advanced prototype.

Bibliography

[1] K. Albrecht and L. McIntyre. *SPYCHIPS: How Major Corporations and Government Plan to Track Your Every Purchase and Watch Your Every Move*, Penguin, 2006.

[2] A. Alkassar, C. Stuble and A. Sadeghi. *Secure Object Identification: or Solving the Chess Grandmaster Problem.* Proceedings of New Security Paradigms Workshop, pp 77–85, 2003.

[3] Analog Devices. *FilterPro – MFB and Sallen-Key Low-Pass Filter Design Program.* `http://focus.ti.com/lit/an/sbfa001a/sbfa001a.pdf`

[4] G. Avoine. *Bibliography on Security and Privacy in RFID Systems.* `http://www.avoine.net/rfid/`

[5] D. Balfanz, D.K. Smetters, P. Stewart and H. Chi Wong. *Talking to Strangers: Authentication in Adhoc Wireless Networks.* Proceedings of Symposium on Network and Distributed Systems Security (NDSS), February 2002.

[6] P. Bahl and V.N. Padmanabhan. *RADAR: an in-building RF-based user location and tracking system.* Proceedings of Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies, pp 775–784, March 2000.

[7] J.E. Bardram, R.E. Kjær and M.Ø. Pedersen. *Context-aware user authentication – Supporting proximity-based login in pervasive computing*, UbiComp 2003, Springer-Verlag LNCS 2864, pp 107–123, 2003.

[8] J. Best. *One billion people to get biometrics and RFID tracking by 2015.* March 2004.
`http://www.silicon.com/research/specialreports/protectingid/0,3800002220,39119660,00.htm`

[9] T. Beth and Y. Desmedt. *Identification Tokens – or: Solving the Chess Grandmaster Problem.* Proceedings of Advances in Cryptology (CRYPTO), Springer-Verlag LNCS 537, pp 169-177, August 1990.

[10] Bluetooth White Paper. 'Simple Pairing'. August 2006.
`http://www.bluetooth.com/Bluetooth/Apply/Technology/Research/Simple_Pairing.htm`

[11] S.C. Bono, M. Green, A. Stubblefield, A. Juels, A.D. Rubin and M. Szydlo, RSA Laboratories *Security Analysis of a Cryptographically-Enabled RFID Device*, Proceeding USENIX Security Symposium, pp 1–16, August 2005.

[12] 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.

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

[14] J. Bringer and H. Chabanne. *On the Wiretap Channel Induced by Noisy Tags.* Proceedings of European Workshop on Security and Privacy in Ad-Hoc and Sensor Networks (ESAS), pp 113–120, 2006.

[15] Bundesamt für Sicherheit in der Informationstechnik. *Advanced Security Mechanisms for Machine Readable Travel Documents Extended Access Control (EAC).* Technical Guideline TR-03110, September 2007.

[16] Bundesamt für Sicherheit in der Informationstechnik. *Security Aspects and Prospective Applications of RFID Systems.* October 2004.
`www.bsi.bund.de/fachthem/rfid/RIKCHA_englisch_Layout.pdf`

[17] L. Bussard. *Trust Establishment Protocols for Communicating Devices.* PhD Thesis, Eurecom-ENST, September 2004.

[18] L. Bussard and W. Bagga. *Distance-Bounding Proof of Knowledge to Avoid Real-Time Attacks.* Proceedings of IFIP International Information Security Conference, pp 223–238, June 2005.

[19] L. Bussard and Y. Roudier. *Embedding distance-bounding protocols within intuitive interactions.* Security in Pervasive Computing: First International Conference, Springer-Verlag LNCS 2802, pp 143–156, March 2004.

[20] S. Čapkun, L. Buttyán and J. Hubaux. *SECTOR: secure tracking of node encounter in multi-hop wireless networks*, Proceedings of ACM Workshop on Security in Ad Hoc and Sensor Networks (SASN), ACM Press, 2003.

[21] S. Čapkun, M.C. Čagalj and M. Srivastava. *Securing localization with hidden and mobile base stations.* Technical Report, NESL, UCLA, 2005.
`www.syssec.ethz.ch/research/TechrepCovert.pdf`

[22] S. Čapkun and J.P. Hubaux. *Secure positioning in wireless networks.* IEEE Journal on Selected Areas in Communications: Special Issue on Security in Wireless Ad Hoc Networks Vol. 24, Issue 2, pp 221–232, 2006.

[23] S. Čapkun, M. Srivastava, M. Čagalj and J. Hubaux. *Securing positioning with covert base stations.* NESL, UCLA Technical Report TR-UCLA-NESL-200503-01, 2005.
`http://lcawww.epfl.ch/capkun/spot/`

[24] Card Technology. *National ID Project Moves China To Head Of The Pack In Radio Frequency Technology.* February 2007.
`http://www.cardtechnology.com/article.html?id=20070221PGYABPF5`

[25] D. Carluccio, T. Kasper and C. Paar. *Implementation Details of a Multi Purpose ISO 14443 RFID-Tool.* Proceedings of Workshop on RFID Security, pp 181–198, July 2006.

[26] D. Carluccio, K. Lemke, and C. Paar. *Electromagnetic side channel analysis of a contactless smart card: first results.* Presented at the ECrypt Workshop on RFID and Lightweight Crypto, 2005.
`http://www.iaik.tu-graz.ac.at/research/krypto/events/`

[27] J.J. Carr. *Practical Antenna Handbook.* McGraw-Hill, 2001.

[28] C.A.S.P.I.A.N – Consumers Against Supermarket Privacy Invasion and Numbering. *Is Big Brother in your grocery cart?.*
`http://www.nocards.org/`

[29] C. Castelluccia and G. Avoine. *Noisy Tags: Pretty Good Key Exchange Protocol for RFID Tags.* Proceedings of International Conference on Smart Card Research and Advanced Applications (CARDIS), Springer-Verlag LNCS 3928, pp 289–299, April 2006.

[30] C. Castellucia and P. Mutaf. *Shake Them Up.* Proceedings of Mobile Systems, Applications and Services (Mobisys), pp 51–64, June 2005.

[31] D. Caswell and P. Debaty. *Creating Web Representations for Places* In Proceedings of Symposium for Handheld and Ubiquitous Computing (HUC 2000), Springer-Verlag LNCS 1927, pp 255-287, September 2000.

[32] CEPT/ERC REC 70-03 relating to the use of short range devices. Annex 9: *Inductive applications.*
`http://www.atcb.com/publicdocs/New-CEPT-70-03-Document.pdf`

[33] H. Chabanne and G. Fumaroli. *Noisy Cryptographic Protocols for Low-Cost RFID Tags.* IEEE Transactions on Information Theory, Vol. 52, No. 8, August 2006

[34] ChipCon CC1000 Single Chip Very Low Power RF Transceiver.
`www.chipcon.com/files/CC1000_Data_Sheet_2_2.pdf`

[35] 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.

[36] CNET News. *Wal-Mart cancels 'smart shelf' trial.* July 2003.
`http://www.news.com/2100-1017_3-1023934.html`

[37] J. Collins. *Dexit Turns RFID Cards into Cash.*
`http://www.rfidjournal.com/article/view/673`

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

[39] Crossbow Technology. Mica2 node, 2006.
`http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICA2_Datasheet.pdf`.

[40] R. Das. *RFID Forecasts 2006 to 2016*. IDTechEx, January 2006.

[41] G. DeJean and D. Kirovski. *RF-DNA: Radio-Frequency Certificates of Authenticity*, 9th International Workshop Cryptographic Hardware and Embedded Systems (CHES 2007), Springer-Verlag LNCS 4727, pp 346–363, September 2007.

[42] Y. Desmedt. *Major security problems with the 'unforgeable' (Feige)-Fiat-Shamir proofs of identity and how to overcome them*. Proceedings of SecuriCom 88, pp 15-17, 1988.

[43] 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.

[44] DHS Emerging Applications and Technology Subcommittee. *The Use of RFID for Human Identification*. May 2006.
`http://www.dhs.gov/`

[45] 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.

[46] ECMA-340. *Near Field Communication Interface and Protocol (NFCIP-1)*, 2nd Edition. December 2004.
`http://www.ecma-international.org/publications/standards/Ecma-340.htm`

[47] EMVCo. *EMV Contactless Communication Protocol Specification v2.0.* August 2007.

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

[49] E-Passport Mock Port of Entry Test. January 2005.
`http://www.epic.org/privacy/us-visit/foia/mockpoe_res.pdf`

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

[51] Federal Information Processing Standards. *Publication 201-1: Personal Identity Verification (PIV) of Federal Employees and Contractors*. March 2006.

[52] M. Feldhofer, S. Dominikus, and J. Wolkerstorfer. *Strong Authentication for RFID Systems Using the AES Algorithm.* 6th International Workshop Cryptographic Hardware and Embedded Systems (CHES 2004), Springer-Verlag LNCS 3156, pp 357–370, August 2004.

[53] T. Finke and H. Kelter. *Radio frequency identification – Abhörmöglichkeiten der Kommunikation zwischen Lesegerät und Transponder am Beispiel eines ISO 14443-Systems*. Bundesamt für Sicherheit in der Informationstechnik, September 2005.
`http://www.bsi.de/fachthem/rfid/whitepaper.htm`

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

[55] K.P. Fishkin and S. Roy. *Enhancing RFID privacy via antenna energy analysis.* Presented at the RFID Privacy Workshop, 2003.

[56] R.J. Fontana, E. Richley and J. Barney. *Commercialization of an ultra wideband precision asset location system.* Proceedings of IEEE Conference on Ultra Wideband Systems and Technologies, pp 369–373, November 2003.

[57] M. Ghavami, L.B. Michael and R. Kohno. *Ultra wideband signals and systems in communication engineering.* Wiley, 2004.

[58] J. Guerrieri and D. Novotny. *HF RFID Eavesdropping and Jamming Tests.* Electromagnetics Division, Electronics and Electrical Engineering Laboratory, National Institute of Standards and Technology, Report No. 818-7-71, 2006.

[59] G.P. Hancke. *Modulating a noisy carrier signal for eavesdropping-resistant HF RFID.* e & i Elektrotechnik und Informationstechnik, Springer, pp 404–408, November 2007.

[60] G.P. Hancke. *Noisy Carrier Modulation for HF RFID.* Proceedings of First International EURASIP Workshop on RFID Technology, pp 63–66, September 2007.

[61] G.P. Hancke. *Practical attacks on proximity identification systems (short paper).* Proceedings of IEEE Symposium on Security and Privacy, pp 328-333, May 2006. `http://www.cl.cam.ac.uk/~gh275/SPPractical.pdf`

[62] G.P. Hancke. *A practical relay attack on ISO 14443 proximity cards.* Project Report, January 2005. `http://www.cl.cam.ac.uk/~gh275/relay.pdf`

[63] G.P Hancke and M.G. Kuhn. *Attacks on Time-of-Flight Distance Bounding Channels.* Proceedings of First ACM Conference on Wireless Network Security (WISEC'08), pp 194–202, March 2008.

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

[65] A. Harter, A. Hopper, P. Steggles, A. Ward and Paul Webster. *The Anatomy of a Context-Aware Application.* Proceedings of Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking, MOBICOM'99, pp 59–68, August 1999.

[66] E. Haselsteiner and K. Breitfuss. *Security in Near Field Communication.* Proceedings of Workshop on RFID Security, pp 3–13, July 2006.

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

[68] P. Horowitz and W. Hill. *The Art of Electronics*, 2nd Edition, Cambridge University Press, 1989.

[69] Y.C. Hu, A. Perrig and D.B. Johnson. *Packet leashes: A defense against wormhole attacks in wireless networks.* Proceedings of INFOCOM, pp 1976–1986, April 2003.

[70] Y.C. Hu, A. Perrig and D.B. Johnson. *Rushing attacks and defense in wireless ad hoc network routing protocols.* Proceedings of ACM Workshop on Wireless Security, pp 30–40, 2003.

[71] 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.

[72] M. Hutter. S. Mangard and M. Feldhofer. *Power and EM Attacks on Passive 13.56 MHz RFID Devices*, 9th International Workshop Cryptographic Hardware and Embedded Systems (CHES 2007), Springer-Verlag LNCS 4727, pp 320–333, September 2007.

[73] ICODE smart label solutions - contactless smart card ICs.
`http://www.nxp.com/products/identification/icode/index.html`

[74] Institute for Prospective Technological Studies. *RFID Technologies: Emerging Issues, Challenges and Policy Options*, Technical Report EUR 22770 EN, 2007.

[75] International Civil Aviation Organization (ICAO). *Document 9303 Machine Readable Travel Documents (MRTD). Part I: Machine Readable Passports.* 2005.

[76] ISO/IEC 7501 *Identification cards – Machine readable travel documents.*

[77] ISO/IEC 9798. *Information Technology – Security techniques – Entity authentication.*

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

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

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

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

[82] A. Juels. *Minimalist cryptography for RFID tags.* International Conference on Security in Communication Networks, Springer-Verlag LNCS 2864, pp 107–123, 2003.

[83] 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.

[84] A. Juels, D. Molnar and D. Wagner. *Security and Privacy Issues in E-passports*, Proceedings of IEEE/CreateNet SecureComm, pp 74–88, 2005.

[85] A. Juels, R. Rivest and M. Szydlo. *The Blocker Tag: Selective Blocking of RFID tags for consumer privacy.* Proceedings of Conference on Computer and Communications Security (CCS), pp 103–111, October 2003.

[86] H. Karl and A Willig. *Protocols and Architectures for Wireless Sensor Networks.* Wiley, 2005.

[87] C. Karlof and D. Wagner. *Secure routing in wireless sensor networks: attacks and countermeasures.* Elsevier Ad Hoc Networks, Vol. 1, pp 293–315, 2003.

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

[89] B. Karp and H.T. Kung. *GPSR: greedy perimeter stateless routing for wireless networks.* Proceedings of MOBICOM 2000, pp 243–254

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

[91] T. Kindberg, K. Zhang, and N. Shankar. *Context Authentication Using Constrained Channels.* Proceedings of IEEE Workshop on Mobile Computing Systems and Applications (WMCSA), pp 14-21, June 2002.

[92] 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.

[93] O. Kömmerling and M. G. Kuhn. *Design Principles for Tamper-Resistant Smartcard Processors.* Proceedings of the USENIX Workshop on Smartcard Technology (Smartcard 99), pp 9–20, May 1999.

[94] J.D. Kraus and R.J. Marhefka. *Antennas: For All Applications.* 3rd Edition, McGraw-Hill, 2001.

[95] B. Krebs. *Leaving Las Vegas: So long DefCon and Blackhat.* The Washington Post, August 2005.
`http://blogs.washingtonpost.com/securityfix/2005/08/both_black_hat_.html`

[96] J. Krumm and E. Horvitz. *LOCADIO: Inferring Motion and Location from Wi-Fi Signal Strengths.* Presented at the First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services, Mobiquitous 2004, August 22–26, 2004.
`http://research.microsoft.com/~horvitz/locadio.pdf`

[97] D. Kügler. *Security Mechanisms of the Biometrically Enhanced (EU) Passport.* Presented at the 2nd International Conference on Security in Pervasive Computing, April 2005.
`http://www.spc-conf.org/2005/slides/SPC_Passport.pdf`

[98] M.G. Kuhn. *An asymmetric security mechanism for navigation signals.* 6th Information Hiding Workshop, Springer-Verlag LNCS 3200, pp 239–252, May 2004.

[99] M.G. Kuhn. *Compromising emanations: Eavesdropping risks of computer displays.* University of Cambridge, Technical Report UCAM-CL-TR-577, December 2003.

[100] J. Landt and B. Catlin. *Shrouds of Time: The history of RFID.* AIM White Paper, October 2001.
`http://www.aimglobal.org/technologies/rfid/resources/shrouds_of_time.pdf`

[101] J. Lettice. *Germany rolls out ePassport II.* November 2007.
`http://www.theregister.co.uk/2007/11/01/german_g2_epassport/`

[102] D. Liu, P. Ning and W. Du. *Attack-resistant location estimation in sensor networks.* Proceedings of IEEE Information Processing in Sensor Networks, pp 99–106, 2005.

[103] D. Liu, P. Ning and W. Du. *Detecting malicious beacon nodes for secure location discovery in wireless sensor networks.* Proceedings of International Conference on Distributed Computing Systems, pp 609–619, 2005.

[104] London Transport Oystercard.
`http://www.oystercard.com`

[105] Mastercard PayPass.
`http://www.paypass.com/`

[106] MAXIM-IC 1471 315MHz/434MHz Low-Power, 3V/5V ASK/FSK Superheterodyne Receiver.
`http://www.maxim-ic.com/quick_view2.cfm/qv_pk/4304`

[107] R. Mayrhofer, M. Hazas, and H. Gellersen. *An Authentication Protocol using Ultrasonic Ranging.* Lancaster University Technical Report Number COMP-002-2006, October 2006.
`http://eis.comp.lancs.ac.uk/index.php?id=361`

[108] R. Mayrhofer and M. Welch. *A Human-Verifiable Authentication Protocol Using Visible Laser Light.* Proceedings of 2nd International Conference on Availability, Reliability and Security (ARES 2007), pp1143–1147, IEEE CS Press, April 2007.

[109] C. Meadows, P. Syverson and L. Chang. *Towards More Efficient Distance Bounding Protocols for Use in Sensor Networks.* Proceedings of Securecomm and Workshops, pp 1–5, August 2006.

[110] MELEXIS MLX90121 13.56MHz RFID Transceiver.
`http://www.melexis.com/ProdMain.aspx?nID=78`

[111] Microchip 16F87X Datasheet.
`ww1.microchip.com/downloads/en/DeviceDoc/30292c.pdf`

[112] Mifare Classic – Information Page.
`http://www.nxp.com/products/identification/mifare/classic/`

[113] Mifare.net.
`http://mifare.net/`

[114] D. Molnar and D. Wagner. *Privacy and Security in Library RFID Issues, Practices, and Architectures.* Proceedings of ACM Conference on Computer and Communications Security, pp 210–219, 2004.

[115] 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.

[116] National Semiconductors. *A Basic Introduction to Filters – Active, Passive, and Switched-Capacitor*. Application Note 779, April 1991.

[117] NFC Forum.
`http://www.nfc-forum.org/`

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

[119] K. Nohl and H. Plötz. *Mifare: Little Security, Despite Obscurity*. 24th Chaos Communication Congress, December 2007.
`http://events.ccc.de/congress/2007/Fahrplan/events/2378.en.html`

[120] Nokia Introduces NFC Phone That Doubles as Credit Card. January 2007.
`http://www.pcmag.com/article2/0,1895,2079922,00.asp`

[121] Y. Niu, M.B. Nejad, H. Tenhunen and L.R. Zheng. *Design of a Digital Baseband Processor for UWB Transceiver on RFID Tag*. Proceedings of Advanced Information Networking and Applications Workshop, pp 358–361, May 2007.

[122] NXP Semiconductors. *MIFARE – Contactless and Dual Interface Smart Card*.
`http://www.nxp.com/products/identification/mifare/index.html`

[123] NXP Semiconductors. *Contactless Reader Components – Data Sheets and Application Notes*.
`www.nxp.com/products/identification/readers/contactless/`

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

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

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

[127] PC World *ACLU's Barry Steinhardt RFID demonstration*. April 2005.
`http://blogs.pcworld.com/staffblog/archives/000609.html`

[128] Philips Semiconductors *Demodulating at 10.7 MHz IF with the SA605/625*. Application Note 1996, October 1997.

[129] J.G. Proakis. *Digital Communications*. 3rd Edition, McGraw-Hill, 1995.

[130] J.G. Proakis and M. Salehi. *Communication Systems Engineering*. Prentice-Hall, 1994.

[131] K.B. Rasmussen and S. Čapkun. *Implications of Radio Fingerprinting on the Security of Sensor Networks*. Proceedings of IEEE SecureComm, 2007.

[132] *rfdump* Project.
`http://www.rfdump.org/`

[133] RFID Gazette. *RFID Technology to Debut at 2006 World Cup Soccer*. February 2006.
`http://www.rfidgazette.org/2006/02/rfid_technology.html`

[134] *rfidiot* Project.
`http://www.rfidiot.org/`

[135] RFID Journal. *The History of RFID Technology*. December 2006.
`http://www.rfidjournal.com/article/articleview/1338/1/129/`

[136] RF Solutions FM Transmitter and Receiver Modules.
`http://www.rfsolutions.co.uk/acatalog/DS069-7.pdf`

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

[138] S. Reid. *'Safest ever' passport is not fit for purpose*. Daily Mail, March 2007.
`http://www.dailymail.co.uk/news/article-440069/`
`Safest-passport-fit-purpose.html`

[139] M. Rieback, B. Crispo, and A. Tanenbaum. *RFID Guardian: A battery powered mobile device for RFID privacy management*, Australasian Conference on Information Security and Privacy – ACISP'05, Springer-Verlag LNC 3574, pp 184–194, 2005.

[140] M.R. Rieback, B. Crispo and A.S. Tanenbaum. *The Evolution of RFID Security*. IEEE Pervasive Computing, Vol. 5, Issue 1, pp 62–69, January 2006.

[141] M.R. Rieback, G.N. Gaydadjiev, B. Crispo, R.F.H. Hofman and A.S. Tannenbaum. *A Platform for RFID Security and Privacy Administration*. Proceedings of Usenix Large Installation System Administration Conference, pp 89–102, December 2006.

[142] M. Roberti, *Fear of Big Brother*, RFID Journal.
`http://www.rfidjournal.com/article/view/276`

[143] H. Robroch. *ePassport Privacy Attack*. Riscure presentation at Cards Asia Singapore, April 2006.
`http://www.riscure.com/2_news/passport.html`

[144] S.E. Sarma, S.A. Weis and D.W. Engels. *RFID Systems, Security & Privacy Implications*, Auto-ID Labs, 2002.
`http://www.autoidlabs.org/whitepapers/`

[145] N. Sastry, U. Shankar and D. Wagner. *Secure verification of location claims*. Proceedings of ACM Workshop on Wireless Security, pp 1–10, 2003.

[146] I. Satoh. *Location-based services in ubiquitous computing environments*, Service-Oriented Computing – ICSOC 2003, Springer-Verlag LNCS 2910, pp 527–542, November 2003.

[147] O. Savry, F. Pebay-Peyroula, F. Dehmas, G. Robert and J. Reverdy. *RFID Noisy Reader How to Prevent from Eavesdropping on the Communication?*, 9th International Workshop Cryptographic Hardware and Embedded Systems (CHES 2007), Springer-Verlag LNCS 4727, pp 334–345, September 2007.

[148] C.E. Shannon. *A Mathematical Theory of Communications.* Bell Systems Technical Journal, Vol. 27, pp 623–656, 1948.

[149] D. Singelée, B. Preneel. *Distance Bounding in Noisy Environments.* European Workshop on Security and Privacy in Ad-Hoc and Sensor Networks (ESAS), Springer-Verlag LNCS 4572, pp 101–115, 2007.

[150] Smart Card Alliance.
`http://www.smartcardalliance.org/`

[151] Smart Card Alliance. *Contactless Technology for Secure Physical Access: Technology and Standards Choices.* Publication No. ID-02002, October 2001.

[152] J. Sorrels. *Optimizing read range in RFID systems*, EDN.
`http://www.edn.com/article/CA84480.html`

[153] F. Stajano and R.J. Anderson. *The Resurrecting Duckling: Security Issues for Ad-Hoc Wireless Networks.* Security Protocols Workshop, Springer-Verlag LNCS 1796, pp 172-194, April 1999.

[154] ST Electronics. *13.56MHz Short Range Contactless Memory Chip 176 bit USER EEPROM and 64 bits Unique ID.*
`http://www.st.com/stonline/products/literature/ds/8747/sr176.pdf\`
`#search=\%22st\%20sr176%22`

[155] ST Electronics. *13.56 MHz Short-Range Contactless Memory Chip with 4096-bit EEPROM, Anti-Collision and Anti-Clone Functions.*
`http://www.st.com/stonline/products/literature/ds/8887/srix4k.pdf`

[156] ST Microelectronics. *How to Extend the Operating Range of the CRX14 Contactless Coupler Chip.* Application Note AN1954, 2005.

[157] H. Stockman. *Communication by Means of Reflected Power.* Proceedings of the IRE, pp 1196-1204, October 1948.

[158] C. Tang and D. O. Wu. *Distance-Bounding Based Defense Against Relay Attacks in Wireless Networks.* IEEE Transactions on Wireless Communications, Vol. 6, No. 11, pp 4071–4078, November 2007.

[159] Texas Instruments. *FilterPro v2.*
`http://focus.ti.com/docs/toolsw/folders/print/filterpro.html`

[160] Texas Instruments. *HF Antenna Cookbook.*
`http://www.ti.com/rfid/docs/manuals/appNotes/HFAntennaCookbook.pdf`

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

[162] Ubisense. White papers and datasheets, 2003–2006. `http://www.ubisense.net`

[163] UK Indymedia. *Gillette Pulls RFID Trial – Campaign Continues.* August 2003. `http://www.indymedia.org.uk/en/2003/08/275818.html`

[164] Visa PayWave. `http://www.visapaywave.co.uk/`

[165] B. Walters and E. Felten. *Proving the location of tamper resistant devices.* February 2003. `ftp://ftp.cs.princeton.edu/techreports/2003/667.pdf`

[166] R. Want, A. Hopper, V. Falcão and J. Gibbons. *The Active Badge Location System.* ACM Transactions on Information Systems, Vo. 10, Issue 1, pp 91–102, 1992.

[167] Wave the Card for Instant Credit. December 2003. `http://www.wired.com/news/technology/0,1282,61603,00.html`

[168] S.A. Weis, S.E. Sarma, R.L. Rivest and D.W. Engels. *Security and Privacy Aspects of Low-Cost Radio Frequency Identification Systems*, First International Conference on Security in Pervasive Computing, Springer-Verlag LNCS , pp 201–212, March 2003.

[169] J. Werb and C. Lanzl. *Designing a positioning system for finding things and people indoors.* IEEE Spectrum, Volume: 35, Issue: 9, pp 71–78, September 1998.

[170] F. Wong and F. Stajano. *Multi-Channel Protocols.* Proceedings of Workshop on Security Protocols, April 2005.

[171] A.D. Wyner. *The Wire-Tap Channel.* Bell Systems Technical Journal, Vol. 54, pp 1355–1387, October 1975.

[172] Xilinx, Inc. *Manchester Encoder-Decoder for Xilinx CPLDs.* Application Note XAPP339 (v1.3), October 2002.

[173] J. Yoshida. *Tests reveal e-passport security flaw.* August 2004. `http://www.eetimes.com/showArticle.jhtml?articleID=\\45400010`

[174] P. Yu, P. Schaumont and D. Ha. *Securing RFID with Ultra-Wideband Modulation.* Proceedings of Workshop on RFID Security (RFIDSec),pp 27–39, July 2006.

[175] R. Zetik, J. Sachs and R. Thome. *UWB localization – active and passive approach.* Proceedings of 21st IEEE Instrumentation and Measurement Technology Conference, Vol. 2, pp 1005–1009, May 2004.

Appendix A

Glossary

AC: Alternating Current
ACLU: American Civil Liberties Union
ADC: Analog-to-Digital Converter
AES: Advanced Encryption Standard
AFI: Application Family Identifier
AIDC: Automatic Identification and Data Capture
AoA: Angle-of-Arrival
APDU: Application Data Unit
ASK: Amplitude-Shift Keying
AWGN: Additive White Gaussian Noise
BCC: UID check byte, calculated as exclusive-or over the 4 previous bytes.
BAC: Basic Access Control
BPSK: Binary Phase-Shift Keying
BSI: Bundesamt für Sicherheit in der Informationstechnik
(German Federal Office for Information Security)
BW: Bandwidth
CAC-C: Common Access Card with Contactless
CID: Card Identifier
CRC: Cyclic Redundancy Check
DB: Distance-Bounding
DC: Direct Current
DES: Data Encryption Standard
DHS: (US) Department of Homeland Security
DPA: Differential Power Analysis
DSP: Digital Signal Processing
EAC: Extended Access Control
ECC: Elliptic Curve Cryptography or Error Correction Code
EEPROM: Electrically Erasable Programmable Read-Only Memory
EER: Equal Error Rate
EGT: Extra Guard Time
EMV: Europay, MasterCard and Visa
EoF: End-of-Frame
EPC: Electronic Product Code
FDT: Frame Delay Time
FFT: Fast Fourier Transform

FIPS: (US) Federal Information Processing Standard
FIR: Finite Impulse Response
FPGA: Field-Programmable Gate Array
FRAC: First Responder Authentication Card
FSK: Frequency-Shift Keying
FWI: Frame Waiting Time Integer
FWT: Frame Waiting Time
FU: Functional Unit
HF: High Frequency (3–30 MHz)
Hz: Hertz
IC: Integrated Circuit
ICAO: International Civil Aviation Organisation
IEC: International Electrotechnical Commision
IF: Intermediate Frequency
IFF: Identification: Friend or Foe
ISO: International Organisation for Standardisation
JCOP: Java Card Operating System
LF: Low Frequency (30–300 kHz)
LM: Location Manager
MAC: Message Authentication Code
MAD: Mutually Authenticated Distance-Bounding
MCU: Micro-Controller Unit
MRTD: Machine Readable Travel Document
MULTOS: Multi-application smart card operating system
NFC: Near-Field Communication
NFCIP: Near-Field Communication Interface and Protocol
NIST: (US) National Institute for Standards and Technology
NKA: NFC Key Agreement
NRZ: Non-Return-to-Zero
NTP: Noisy Tag Protocol
NVB: Number of Valid Bits
OCR: Optical Character Recognition
OEM: Original Equipment Manufacturer
OOK: On-Off Keying
PCB: Printed Circuit Board
PCD: Proximity Coupling Device
PDA: Personal Digital Assistant
PICC: Proximity Integrated Circuit Card
PIV: Personal Identity Verification
PLL: Phase-Locked Loop
PRN: Pseudo-Random Noise
PSK: Phase-Shift Keying
PPM: Pulse-Position Modulation
RAM: Random Access Memory
RFID: Radio Frequency Identification
RSA: Rivest, Shamir, Adleman
RSS: Received-Signal-Strength
RTT: Round-Trip-Time

SNR: Signal-to-Noise Ratio
SoF: Start-of-Frame
TfL: Transport for London
ToF: Time-of-Flight
TWIC: Transportation Worker Identification Credential
UHF: Ultra-High Frequency (0.3–3 GHz)
UID: Unique Identifier
UWB: Ultra-Wideband
WEP: Wireless Equivalency Privacy