Security and privacy analysis of Radio Frequency Identification systems

Mahshid Yassaei

Master of Applied Science

Department of Computer Science

McGill University

Montreal,Quebec

July 2012

A thesis submitted to McGill University in partial fulfilment of the requirement for the degree of Master of science

DEDICATION

To my beloved parents, Shiva and Abbas

and to my ever smiling friend Hesam...

ACKNOWLEDGEMENTS

I cannot find words to express my gratitude to my co-advisors, Prof. Claude Crepeau and Prof. Jose M. Fernandez, for their trust on me in the first place and their kind collaboration and valuable comments during this study. I wish to thank Dr. Carlton Davis and Mr. Pier-Luc St.Onge who helped me a lot during this project, as well.

I should also thank Mr.Mathieu Petitpas and Mr. Pierre Brun-Murol for the French translation of the abstract.

This thesis would have not been possible without the kind support of my parents, Shiva and Abbas who never stopped believing in me.

ABSTRACT

Radio Frequency Identification (RFID) technology is widely used for various applications from access control to object tracking systems. Automation and faster services provided by this technology have striking effects on our daily life. However, there are several security and privacy concerns about RFID systems that remain unsolved. During the past years, several attacks have been designed against Mifare Classic and HID iClass, two of the most widely used RFID systems on the market.

The aim of this study was to improve the security and privacy mechanisms of RFID systems through the development of tools and the methodology of system analysis, in the hope to find the possible flaws before the adversaries do. As an example, efforts were made to partially analyze OPUS cards (the RFID-enabled public transportation passes in Montreal) and several security and privacy violating specifications of these cards were highlighted. It was revealed that the static identification number of the card is transfered in the anticollision process which can be used to track the card holder without his consent. In addition, the information about the last three usages of the card (the time, the date and the metro/bus station) are transferred unencrypted and before the authentication process. Only a linear conversion is applied to the information which can be reversed by a simple application such as the one developed and provided in this study.

Furthermore, design modifications to improve the security and privacy level of RFID systems were provided. These modifications are categorized based on the cost and

the disruption of service that the application of these modifications imposes to the manufacturing company.

Key Words: RFID Systems, Privacy, Security, OPUS Cards

ABRÉGÉ

Les technologies de radio identification (RFID) sont fortement utilisées dans diverses applications qui vont du contrôle d'accès aux systèmes de traçabilité d'objets. L'automatisation et la rapidité accrue des services que ces technologies rendent possibles ont des effets marqués sur notre vie quotidienne. Cependant, les systèmes RFID comportent de nombreux problèmes de sécurité et de protection de la vie privée qui ne sont toujours pas résolus. Au cours des dernières années, de nombreuses attaques ont été conues contre la puce Classic de MIFARE ainsi que la puce iClass d'HID, deux des systèmes RFID les plus répandus sur le marché. Le but de cette étude est d'améliorer les mécanismes de sécurité et de protection de la vie privée des systèmes RFID par le développement d'outils et la méthodologie d'analyse des systèmes, dans l'espoir de découvrir les failles de sécurité potentielles avant que des adversaires ne le fassent. Par exemple, nous avons procédé à une analyse partielle des cartes OPUS (les cartes qui contiennent les titres de transport en commun utilisés à Montréal, qui font usage de la technologie RFID), et mis en évidence de nombreux éléments des spécifications de ces cartes qui représentent une faille de sécurité ou de protection de la vie privée. Nous avons découvert que le numéro d'identification statique de la carte est transmis durant le processus anticollision, ce qui peut être utilisé pour suivre la trace du détenteur de la carte sans son consentement. De plus, des informations concernant les trois dernières utilisations d'une carte (l'heure, la date, et la station de métro ou d'autobus) sont transmis sans être chiffrés, et avant le processus d'authentification n'ait lieu. Seule une conversion linéaire est appliquée sur

l'information, et cette conversion peut être inversée par une simple application telle que celle que nous avons développé au cours de cette étude.

De plus, nous présentons des modifications visant à améliorer le niveau de sécurité et de protection de la vie privée des systèmes RFID. Nous classons ces modifications sur la base de leur coût et de la gravité des interruptions de service que l'application de ces modifications ferait subir au manufacturier.

Mots clés: Systèmes RFID, protection de la vie privée, sécurité, cartes OPUS

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
Introduction to RFID Systems

Radio Frequency Identification (RFID) is the term used for any identification system in which an electronic device is attached to an object and communicates with a base station through radio frequency (RF) signals.

The electronic device that is attached to the object is usually called a tag. A tag is capable of receiving and transmitting RF signals, performing simple computations and storing very limited amount of data.

A reader is an electronic device that can recognize the tag and communicate with it through RF signals. A reader is connected to a backend system that runs the applications and stores the data.

The tag, the reader and the backend system are the main components of an RFID systems. However, in some cases, there is also a middleware between the reader and the backend system to preprocess the data before sending it to the backend system. The communication between tag and reader is through RF signals, unlike the one between reader and backend system/middleware which is through a secure wired channel (Figure 1-1).

Much of the interest in RFID systems is driven by the interest of some government agencies such as U.S Department of Defense (DoD), or Food and Drug Administration (FDA). However, big private companies such as WalMart also played a part by using RFID tags on pallets shipped to its stores since early 2005 [1].

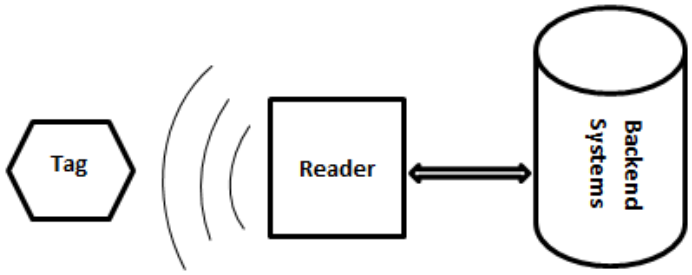

Figure 1–1: Components of an RFID System

Although RFID technology was initially used in inventory management only, there are many applications in which this technology is used today. These applications can be categorized as: access control, objects tracking and electronic payment. For example, RFID technology is very regulary used in the access control to office buildings or to the parking or other amenities of the buildings. Other than that, new generation of credit cards and payment systems use RFID technology for a faster and more automatic way of payment. RFID technology is also used to track different kind of objects or livestocks like the goods in travel from the factories to the store, or the animals that are in danger of extinction. Each of these applications requires different hardware and software specifications that address the particular need of the system. In spite of the pervasiveness of RFID technology, there are some privacy and security questions related to these systems that are still unresolved. Man-in-the-middle, cryptanalysis, replay, denial of service and cloning attacks are the most common threats against RFID systems. In the case of a poor security and privacy preserving RFID-based access control system, an adversary may be able to make a copy of an

employee's card to get an unauthorized access to the building of a company, or prevent the employees from entering the building by a denial of service attack. In the case of a payment system, the adversary might be able to extract information about the card holder's bank account or to use the information for further payments from the card holder's account without his consent. Due to strict limitations of power and computation, RFID systems cannot use classical cryptographic algorithms for data protection. Serious frauds have been found in *proprietary* data protection algorithms that are used in RFID systems. Therefore, the challenge of designing and developing a secure and privacy preserving RFID system is a very interesting problem which merits much investigation.

At the same time, developing the methodologies for security/privacy analysis of existing RFID systems helps the designers to detect the vulnerabilities in the systems before the adversaries do. They will have the opportunity to modify their system with respect to the detected vulnerabilities to prevent any serious attack with malicious intents.

The solutions to the security/privacy problems of the existing RFID systems can be categorized into five groups:

- software updates
- hardware updates
- card-type changes (the same technology)
- new technology
- change of policy

from the cheapest type of modifications which is software updates to more expensive updates such as replacing the whole system by a new technology, or a change of policy in the way the RFID systems are designed and manufactured. In this work we aim to find solutions that address the security problems of these systems, while minimizing the cost and disruption of service for the systems in use.

During this thesis, our research questions were the followings:

1. Can we find any new type of vulnerability in the systems already analyzed by the previous research groups? Or can we find the same type of vulnerability in the systems that have not been analyzed before?
2. What is the methodology for security analysis of an RFID system? Can we follow this methodology to analyze a system that has not been studied before?
3. What are the modifications we may apply to current RFID systems to make them more secure?

To address these questions, we studied the previous attacks that were launched against the TI DST, Mifare Classic and HID iClass RFID cards, in order to study the methodology of the attacks and the vulnerabilities that were found about the cards which is presented in Chapter 2. In the same chapter, we study the authentication algorithms proposed by different research groups to show their inefficiency and vulnerabilities that have been discovered through the attacks designed against them very soon after. In Chapter 3, we introduce our methodology for security analysis of an RFID system and we follow this methodology to analyze the OPUS card (RFID card used for transportation in the city of Montreal). The results of our analysis along with appropriate solutions are discussed in this chapter. In the last chapter,

the thesis is concluded and our practical solutions are proposed. The solutions are categorized into five levels of updates as discussed above. These solutions are based on our knowledge of constraints and challenges and are likely to be found helpful for the improvement of the next generation of RFID systems to come.

CHAPTER 2
RFID Technology

In this chapter we present background information related to RFID systems. We start by outlining various means of classification for RFID systems, next we detail the communication scheme between a tag and a reader, the applications of this technology and the security and privacy threats to them. Following this, the chapter continues with a survey of previous attacks against RFID systems during the past years, and it concludes by summarizing the proposed solutions to the problem of data protection in RFID systems.

2.1 Classification of RFID tags

RFID tags are classified based on the power, computation or memory constraints they exhibit.

In terms of power resources, RFID tags are categorized into 3 groups of passive, semi-passive and active tags:

- Passive RFID tags do not have any power source; instead they use the magnetic field of the reader to perform computations and for sending and receiving data. The communication range of these tags is relatively short (3 meters or less [2]) and the computation power is just enough for bit-wise operations and some simple symmetric key cryptographic computations. Passive tags have small data storage capacity of only around 128 bytes on their chip.

- Semi-passive tags have a small source of power that can be used in combination with the energy from the magnetic field of the reader to complete the tasks. They have higher computation and communication capacities and can handle computationally intensive cryptographic functions.
- Active RFID tags have their own power sources and do not depend on the readers magnetic field. In spite of the other two types, these type of RFID tags can initiate the communication and can be used in a wireless ad-hoc network with other tags. They have computational and power resources to support stronger security mechanisms to protect the tag against malicious behaviors.

The other method of classifying RFID tags is based on Juels's classification scheme [5] which classifies RFID tags according to their computational resources. In this scheme, tags are classified either as basic tags or symmetric key tags:

- Basic tags have almost no computational capacity. These tags cannot handle cryptographic computations because of the small number of gates available on the chip which is just enough to store the unique serial number of the item they are attached to.
- The symmetric-key tags, on the other hand, are those which have computational capacity that is enough for symmetric-key cryptographic functions. Symmetric-key tags can devote up to 4K gates to security related computations [35].

There is also another method of classifying RFID tags based on the type and/or amount of memory on the tags [3]. This classification characterizes RFID tags into 6 different classes:

- Class 0 - read-only Memory, preprogrammed passive tags.
- Class 1 - write once read many memory (WORM).
- Class 2 - Passive read-write tags that can be written to at any point during or after they are manufactured.
- Class 3 - Read-write with sensors capable of recording parameters like temperature, pressure, and motion; the tag can either be semi-passive or active.
- Class 4 - Read-write active tags that can communicate with other tags and readers.
- Class 5 - Similar to Class 4 tags but with additional functionality; tags in this class can provide power to other tags and communicate with devices other than readers.

In this thesis, we focus mainly on passive, symmetric-key, class 2 RFID tags because a) they are more prevalent, and b) there are more security concerns regarding them.

2.2 Tag-Reader Communication

In most RFID systems, communication between a reader and a tag has 3 phases. The first phase is the anti-collision loop where the reader selects the tag it wishes to communicate with. This process is needed to avoid collision, when there is more than one tag in the magnetic field of the reader; secondly, a mutual authentication process between the reader and tag is intended to verify the authenticity of the parties involved in the communication; and finally the main phase involves exchange of encrypted messages through the RF channel (Figure 2-1).

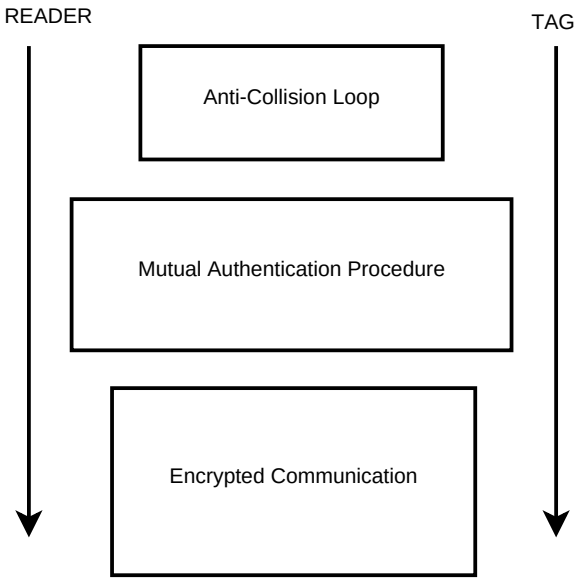

Figure 2–1: communication phases between a reader and a tag

Following is a non-exhaustive list of major organizations that have produced standards for RFID systems.

- ANSI (American National Standards Institute)
- ISO (International Organization for Standardization)
- EPCglobal (Electronic Product Code global)
- CEN (Commission for European Normalization)

We had the opportunity to work with systems based on three standards: ISO 15693, ISO 14443a and ISO 14443b. These standards currently consist of the following four parts:

- Part 1. Physical characteristics
- Part 2. Radio frequency power and signal interface.
- Part 3. Initialization and anti-collision.

- Part 4. Transmission protocol.

An RFID system that follows any of these protocols usually follows every details of the protocol according to the standard. But in some situations the vendors may decide to change some parts of the transmission protocol, possibly as a security decision to mislead the adversaries, or due to other engineering constraints.

2.3 RFID Applications

There are many industrial RFID applications. Excessive needs of automation and speed in industry, motivated different sectors of industry to use RFID systems. Some of the most important RFID applications are the following:

- **Inventory management**: RFID technology is widely used in large retail stores' inventory systems, such as Walmart. Tags are affixed to goods when packed at the factory to track the packages on their way to the truck, on the boat, entering the supply chain and through distribution until they reach their final in-store destinations. They can also be used in "smart shelves" to track all the merchandize and alert the personnel if the items are misplaced or stolen from the store. RFID tags that are used in these kind of applications are usually very cheap, around 5 cents each and the main security threat against them is Denial of Service attack which disables the tags from further communications [6].
- **Animal tracking**: Equipping pets, livestock, exotic animals and endangered species with RFID tags helps in the process of tracking, recovering and managing them. For example, RFID tags with GPS can be used to track the behavior of endangered animals, to help protect them. This method has been applied to

protect dolphins [6]. Tags used in animal tracking are usually more expensive. They might also be required to operate on their own source of power and to support a longer range of communication. Since the livestocks are usually more expensive and more important than the regular objects in inventory management, the RFID tags that are used in these applications should be equipped with more strict security and privacy preserving mechanisms.

- **Access control systems**: RFID technology can be used in access control systems of buildings, libraries or companies to guarantee the security of these places [9]. RFID tags can also be used in the ignition key of automobiles. The latter is widely credited for reducing auto theft by 50% [10]. These applications usually require a passive RFID tag of short range, around 5 cm, with more security mechanisms that supports mutual authentication and encryption of the data in transfer. RFID tags in these applications should be more specifically protected against cloning attacks in which the goal of the adversary is to make a copy of the card or the reader.
- **Payment systems**: Speedpass, introduced in 1997, is probably one of the best known instances of using RFID systems for payment [6]. Other examples include new generation of credit cards that use RFID technology and the prepaid cards that are used for access control to public transportation. RFID tags in payment systems are usually passive, short range and low frequency. Relatively strong authentication and encryption schemes are usually used to protect the user's financial information stored on the tag, or to prevent the reader or the tag from cloning attacks.

- **E-passport**: RFID tags can be embedded in passports to record and transmit the holders biometric information, including fingerprints. In comparison to traditional passports with just a photo ID, counterfeiting an electronic passport can be much harder if the RFID technology can be trusted. RFID tags in this application usually follow the ISO 14443 standard and are protected with a long enough (usually 128 bits) key, through a trusted cryptographic function [64]. A cloning attack is the most dangerous threat against this application.

2.4 Security and Privacy Challenges in an RFID System

In general, privacy issues are related to information leakage that happens without the consent of the owner of the system. In the current context, we may want to distinguish data privacy which concerns the privacy of the data stored in the memory of the tag, from location privacy, which is related to data that can be used to track the physical location of the tag or the tag holder.

Security, on the other hand, concerns an adversary being able to interfere in the system, take advantage of the service without being authorized to, or make the system unavailable for authorized users.

Security and privacy threats can be categorized according to the adversary's power or the goals of the attack.

2.4.1 The adversary's power

Five levels of power can be assumed for an adversary [47]:

- The most powerful adversary is the one who can have physical access to the system and do some laboratory experiments on the reader or the tag. The attackers in [41] and [42] are examples of this category. The attacker can probe

the device or even remove materials from the tag's or the reader's chip, through physical or chemical methods.

- The second most powerful attacker can not have physical access to the system, but can actively participate in a legitimate communication between a reader and a tag, by spoofing either of the parties or manipulate the signals in transfer. Man-in-the-middle attacks belong to this category.
- The third group are the passive attackers who can only eavesdrop the communications between a reader and a tag. Therefore, the input to the attacks of this group is the stream of '0's and '1's from a legitimate communication. Passive cryptanalysis attacks like [16] is an example of this type of attack.
- The fourth group cannot even eavesdrop the messages. The attackers of this group can detect the presence of a message but not a clear "logical" stream as an input. The inputs to the attacks of this category are hence restricted to some traffic analysis and information about when and how many messages have been transferred between the two parties.
- The weakest attacker cannot even detect the presence of a message. This category's attacks is restricted to disrupting broadcasts or frequency jamming which is a type of Denial of Service attack.

2.4.2 The goals of the attack

The objectives of an attack can be any or a combination of the following:

- Tracking the tag or the tag holder
- Reading the content of the tag's memory
- Writing on the tag or manipulating the information of the tag or the reader

- Interfering in the communication through manipulation of the messages' contents
- Disabling a reader or a tag from further communications (Denial of Service attacks)
- Cloning a reader or a tag
  - Partial break, which means being able to clone only one specific RFID card
  - Full break, which means being able to clone any card given its non-secret information, such as its UID

The adversary may try different techniques to reach these objectives. A non-exhaustive list of these techniques is described here.

2.4.3 Attack techniques

UID tracking, skimming and eavesdropping are the most common techniques of attack against RFID systems, through which the information is gathered about the system to be used in further analysis. There are also other techniques through which a denial of service attack is launched. These techniques are discussed in more detail here.

UID Tracking

Most RFID tags (including the ones discussed in this thesis) transmit their UID (unique identification number) through the air in response to a "hello" message from a reader in the anticollision process. Although this ID might be random and might not carry any private information related to the card holder, the uniqueness of the ID makes the card holder vulnerable against clandestine tracking. Marketers

can use this unique ID to create a user profile of their customers and use it when they return to the market, or for aimed advertisement [5].

Skimming and Eavesdropping attacks

Another type of attack against an RFID system is the one in which the adversary tries to take advantage of wireless communication in the system and read or write on the memory. The two common attacks in this category are skimming and eavesdropping.

In a skimming attack, a tag is queried by RF signals in its vicinity, to access or modify the data on the memory of tag. In these attacks, an adversary can control where and when the attack is performed. Actually, the attacker's challenge, in practice, is to increase the operational range of the system by powering the tag to communicate with the attackers reader, over a greater distance.

There are two types of distances involved in this attack:

- The distance at which the attacker can power and query the tag.
- The distance at which the attacker can power, query the tag and recover the response from it.

In eavesdropping attack, on the other hand, the attacker sniffs the communication between a reader and a tag in silence, but does not interfere with the RF signals of its own. As a result, he can launch the attack at a longer distance than what is required for skimming attacks. But the attacker is limited as to when and where he can launch the attack, because he needs to be ready to attack during the time window when the tag is communicating with a legitimate reader. The forward channel, which is the channel from the reader to the tag, has a longer range and is more vulnerable

against this attack in comparison to the backward channel which is the channel from the tag to the reader. There are 3 difference distances involved in this attack:

- The distance at which the attacker can detect the communication but cannot recover the actual data.
- The distance at which the attacker can recover the messages from the reader but not the responses from the tag.
- The distance at which the attacker can recover the data from both the reader and the tag.

A successful eavesdropping attack is usually is capable of recovering data from both channels.

Figure 2-2 shows the difference between skimming and eavesdropping attacks [33]. While there have been some research on how to limit these different types of distances, there are still questions to be answered about how far skimming and eavesdropping distances can be performed.

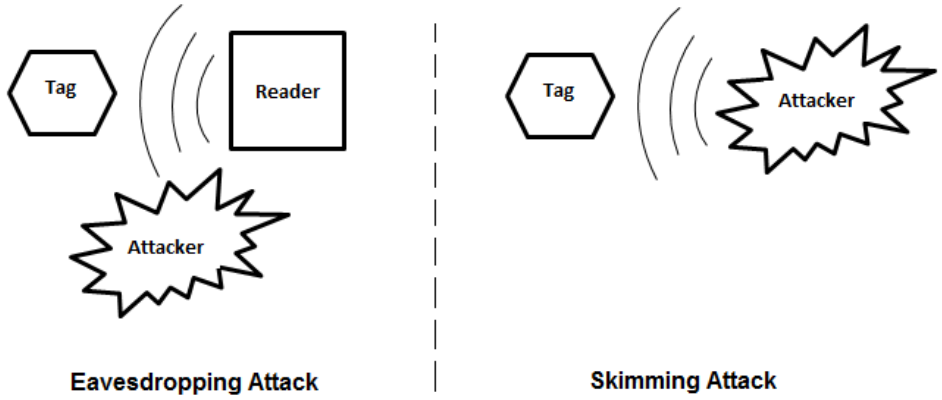

Figure 2–2: Difference between skimming and eavesdropping attacks

Denial of Service Attacks

Another threat against an RFID system is a de-synchronization attack. In this attack, the adversary destroys the synchronization between a reader and a tag and thus disables the authentication capability of the RFID tag. This can lead to a Denial of Service (DoS) type of attack. The other possible DoS attack is the kill command attack. A kill command is used by the manufacturer to kill the tag, whenever it is necessary. Each tag is given a password at the time of manufacturing which allows the owner to access the system in order to kill the tag. Due to the limited memory, this password can be easily cracked with a brute force technique. This password is then used by the adversary to kill the tag and make the service unavailable to legitimate users.

Any RFID system can easily be disturbed with frequency jamming which is another type of DoS attack that can be used against the availability of an RFID reader or tag.

2.5 Previous attacks against RFID systems

Traditionally, the confidence level assigned to a cryptographic protocol is determined by two means: a) a formal proof which supports the claimed level of security and, b) the fact that the algorithm has been around for decades and no serious attack has found against it.

In the case of RFID systems, the second approach is not feasible for now, since RFID systems are relatively young and the proposed algorithms cannot be trusted just because no weakness has been found about them yet. However, the first approach can be taken more seriously, and it has started to appear in several research plans such

as [44]. At this point, there is no protocol specifically designed for RFID systems that is formally proven to be secure.

During the past years, many authentication algorithms have been proposed for RFID systems. Some of these algorithms are designed specifically for RFID systems and some are modified versions of modern cryptographic algorithms compatible with the limitations of RFID systems.

The attempt to introduce authentication algorithms for RFID systems has led to a large group of proposed algorithms [37] for most of which vulnerabilities have been found within a year. The manufacturers of RFID systems, however, prefer to use obscurity as a mean of providing security i.e. they hope that by not revealing the encryption/decryption algorithms, methods to defeat the systems cannot be found. As we will see, they have not succeeded to keep their systems secure through this approach, since it is hard to hide the logic of the algorithms when it is implemented on both the card and the reader, and the adversary has access to these devices.

This subsection contains a description of some notorious attacks on these RFID technologies including Digital Signature Transponder, Mifare Classic and HID iClass.

2.5.1 Digital Signature Transponder

The Texas Instruments Digital Signature Transponder (DST) is an RFID system used in variety of application. DST devices helped securing millions of ExxonMobile SpeedPass payment transponders and automobile ignition keys (Figure 2-3). In 2005, a group of researchers at John Hopkins university analyzed DST [54]. This research showed how an adversary can make a copy of the RFID tag (cloning-partial break), manufactured by this company, and take advantage of it. The analysis of this work

consists of three phases: Reverse engineering, key cracking and simulation.

In the reverse engineering phase, they managed to determine the complete functional

Figure 2–3: At left, an ExxonMobile SpeedPass and at right a DST-based automobile ignition key [34]

details of the cipher, by using only oracle or black box access to an ordinary DST. In the key cracking phase, they manage to complete a brute force attack against the only 40-bit key of DST with the help of an array of 16 FUGA operations in parallel, in less than an hour. Later in the simulation phase, they used the key to actually spoof a reader and launch real-world attacks against an ignition key and a Speedups RFID card.

The information that was available at the time of research was:

- the length of the DST cryptographic secret key which was only 40 bits and the fact that it is field-programmable via RF commands in the memory of the tag, meaning that the key can be modified after manufacturing by the user.
- a very vague description of the communication as detailed in the fact that the tag emits a factory-set 24-bit identifier in response to "hello" message. The reader starts the authentication by sending a 40-bit challenge. The tag encrypts the challenge with its key and truncating the resulting ciphertext, returns a 24-bit response.

- a rough schematic of the algorithm (Figure 2-4) that was published in a paper co-authored by Zurich Kaiser (a Texas Instruments employee). The arrangement of the functional components of the cipher are clear, although the logic and the interconnections are lacking [4].

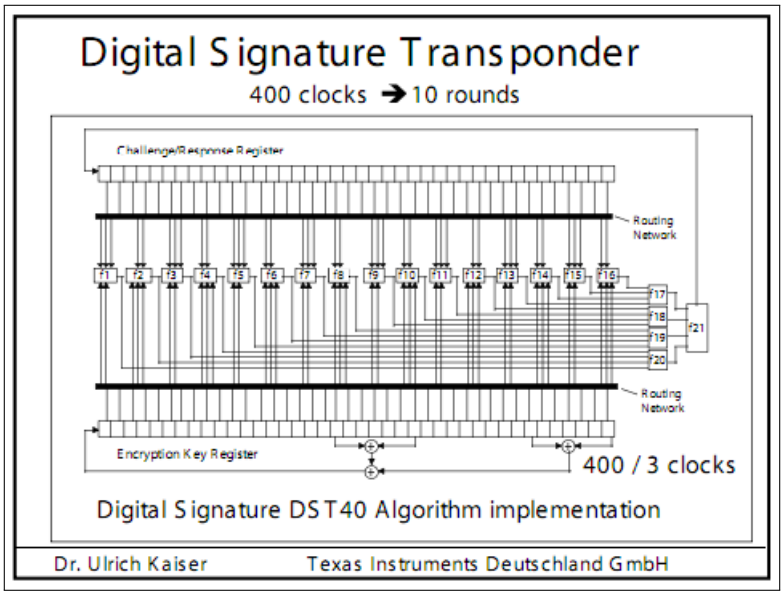

Figure 2–4: Schematic of Kaiser cipher

The researchers reverse engineered a DST device by checking the logical output of the device or essentially a black box attack against it. They used a TI Series 2000-LF RFID Evaluation Kit [14], which had a reader and an antenna for communicating with low frequency RFID systems. Through trial and error they determined the complete functional details of the cipher. They showed several weaknesses in the cipher such as biased challenge register on the reader side, but they did not manage to cryptanalyse the protocol completely.

This was an example of an active scanning attack against an RFID system, that

was feasible mainly because of the short key length. A 40 bits key is very short for the current computational power. Only 2 arbitrary challenge-response pairs plus an array of 16 FUGA can crack the key in less than an hour.

2.5.2 Mifare Classic

Mifare Classic is a product of NXP Semiconductors (formerly Philips) and was the most used RFID tag in the world in 2008, the year that two practical attacks designed against it were published [28]. Mifare Classic is *still* used mainly in payment and transportation systems, providing mutual authentication and data secrecy by means of the CRYPTO1 stream cipher. CRYPTO1 is a proprietary encryption algorithm designed by NP and its design was kept secret. As is shown in Figure 2-5, the stream cipher consists of a linear feedback register (LOF) and a filter function f(·). The function is composed of six instantiations of three smaller functions, which are shown in Figure 2-6.

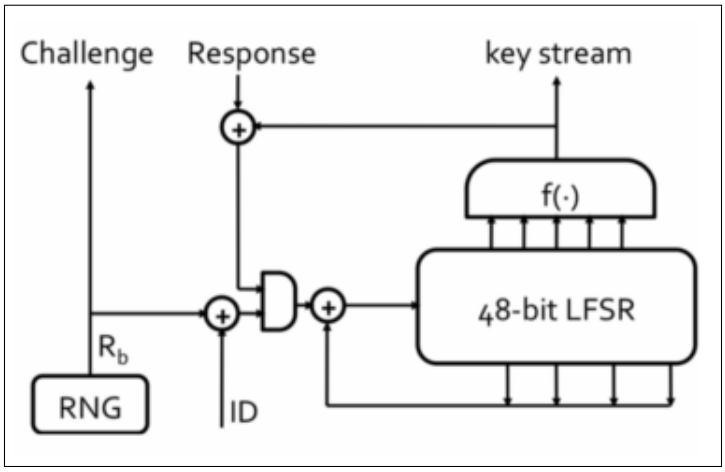

Figure 2–5: CRYPTO1 algorithm

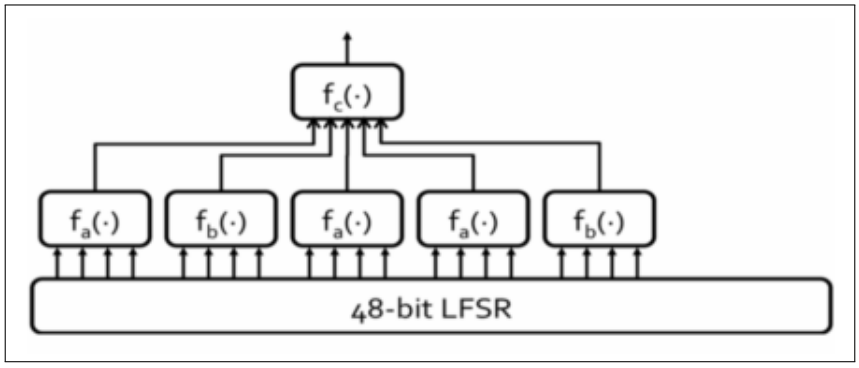

Figure 2–6: Structure of CRYPTO1 filter function

Data on Mifare Classic cards is divided into sectors. Each sector holds two different keys for different access rights (read/write). This division allows various applications to store their encrypted data on private sectors (Figure 2-6).

Two serious attacks were launched against Mifare Classic in 2008, one by researchers

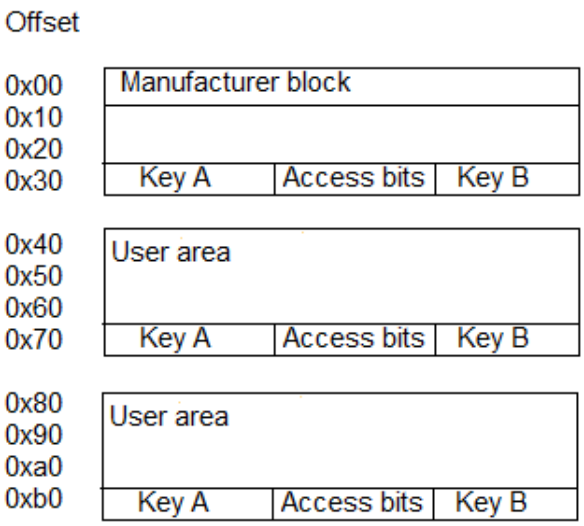

Figure 2–7: Memory layout of Mifare Classic

at the University of Virginia [31] and the other by researchers at Radboud University [28].

The research conducted by Nohl and Evans [31] was a combination of hardware and protocol analysis of the system. The main contribution of the article [31] was to realize that the algorithm was much weaker than the 48-bit key length as a result of a number of design flaws, and to demonstrate that the hardware analysis is not as expensive as it was assumed to be, and can be done in an automated fashion.

The hardware analysis was started by removing the plastic on the chip using acetone. Once they had isolated the silicon chip, they removed each layer (6 layers in total) through mechanical polishing, which led to the circuit in transistor layer. From the images taken of the transistors using a microscope, they built a template matching function which could recognize the gates in the circuit from their transistor combinations.

They knew that the cipher would have to include at least one 48-bit register (for the key) and a number of XOR gates. They managed to detect this combination plus a circuit which appeared to be pseudo-random number generator from the images.

However, from the hardware analysis they could not figure out which inputs were shifted into the cipher and what the order of the bits was. Therefore, they tried to analyze the protocol in order to complete the attack. Different combinations of keys were tested, with single bit changes in the ID, and surprisingly some of the combinations authenticated the tag successfully. Analyzing these combinations, they could derive not just the order of inputs but also the structure of the linear feedback

shift register. Furthermore, They realized that the shift register repeats the pseudo-random sequence after $(2^{48} - 1)$ steps, which could help the attacker pre-compute a code-book to reduce the attack's running time from weeks to minutes.

The main known vulnerabilities of this RFID system were recognized as:

- the update rules of the shift register missing any non-linearity, which can be considered as a serious weakness with our current understanding of block ciphers.
- the values of the register wrapping around every 0.6 seconds after generating all 65,535 outputs which gives the attacker the power to control the generated numbers by controlling the timing.
- the generating LFSR being reset to a known state every time the tag starts operating, which is completely unnecessary and destroys the randomness remaining from the previous transactions and unpredictable noise.
- the pseudo random number generator having the same weaknesses on the reader side, which means that true randomness is lacking from both the reader and the tag sides.

2.5.3 HID iClass

iClass is a technology from HID Global company, licensed from Inside Technologies [58], which was introduced in 2002 as a secure replacement for HID Prox Cards that had no cryptographic function at all. iClass keeps ISO-15693 standard and works at 13.56 MHz. According to the manufacturer, more than 300 millions of these cards have been sold so far. HID iClass is widely used in access control applications, all over the world.

To compete with other products of the market, iClass tags are characterized by two storage capacities: 2Kbit and 16Kbit, with respectively 256 and 4096 bytes of memory. The memory is divided into blocks of 8 bytes (Figure 2-8).

Memory blocks 0,1,2 and 5 are publicly accessible and contain the card serial number (CSN) which is used in the anticollision process, configuration bits, the card challenge c_c and issuer information, respectively. Blocks 3 and 4 contain two diversified cryptographic keys which are derived from two different HID master keys (debit key k_d and credit key k_c) and the id of the tag. The card only stores the diversified keys (kc_{id} and kd_{id}). The remaining blocks are divided into two areas or applications [55].

The first application of an iClass card, HID application, stores the identifier, PIN code, password and other access control information. Read/Write to the HID application requires a valid mutual authentication using a proprietary algorithm that proves the knowledge of kd_{id}.

The second application is defined by the user and is secured by a key kc_{id} derived from k_c.

iClass cards are either "Standard Security" or "High Security". Standard security means that two common keys of k_c and k_d are shared across all HID readers in that mode[1] . Therefore, breaking a single reader once, an attacker can enter all standard

[1] Readers need to store the keys to be able to encrypt/decrypt the messages while communicating with a specific card

security reader and card systems. However, in high security mode different site specific keys are used for different readers. Refer to Figure 2-9 for detailed information about how the keys are generated in each case.

Block	Content	Denoted by
0	Card serial number	Identifier *id*
1	Configuration	
2	e-Purse	Card challenge c_C
3	Key for application 1	Debit key kd_{id}
4	Key for application 2	Credit key kc_{id}
5	Application issuer area	
6...18	Application 1	HID application a_{HID}
19...n	Application 2	$n = 16x - 1$ for xKS

Figure 2–8: HID iClass Memory layout [55]

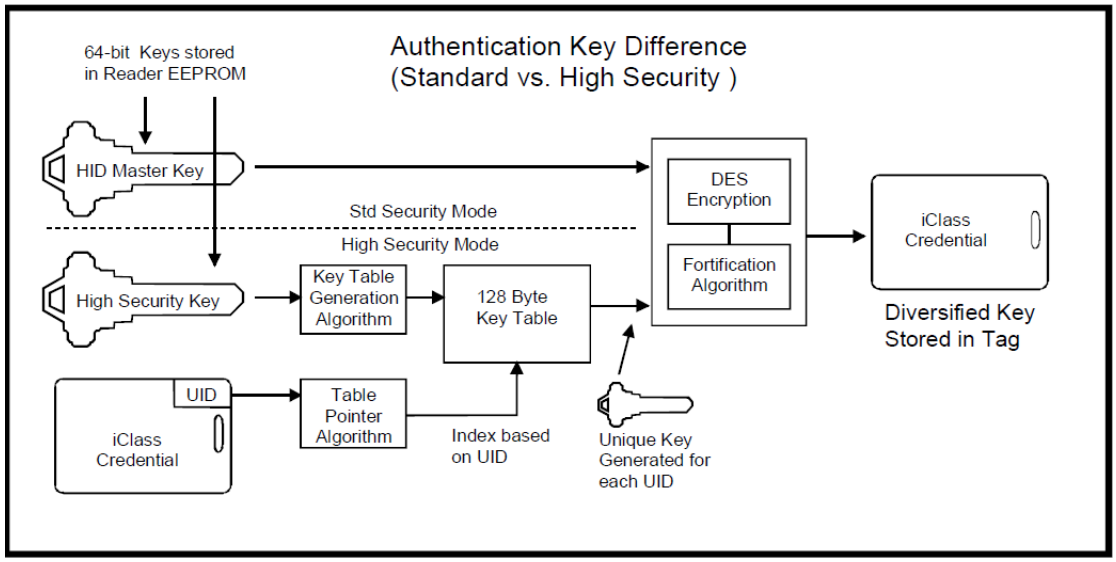

Figure 2–9: Standard security mode vs. high security mode [53]

Three attacks have been designed against HID iClass. The first one, in 2010, was based on hardware analysis of an iClass reader, through which the attacker managed to discover the shared key among all iClass cards/readers in Standard mode [60]. The second attack was against the key diversification algorithm in iClass systems, in August 2011 [55] and finally the last attack was the reverse engineering of the INCrypt32 (which is a cryptographic algorithm for mutual authentication) with the aim of revealing the specifications and designing a cryptanalysis attack against it [19]. In December 2010, Meriac and Plotz showed a procedure to read out the EEPROM of a PIC microcontroller (the case of iClass readers) to find out the shared keys and break all iClass readers in the standard mode [60]. They bought several RW400 readers of the type $6121AKN0000$ and opened one to see the hardware configurations inside. They managed to read the memory of the tag and identify the key from the other bytes with a trial and error method. They could copy an HID iClass card (cloning, partial break) in Standard mode without really being able to decrypt 3DES encrypted content. Figure 2-10

Later in August 2011, an attack against iClass smart cards was reported by Garcia, Koning Gans and Verdult [55]. The authors mainly showed the reverse engineering process of the built-in key diversification algorithm. Figure 2-11 shows the process of diversifying kc_{id}/kd_{id} from k_c/k_d. As the picture shows, the designers of HID iClass have decided to protect the master key by a *single DES* application and use the output C as an input to a diversification algorithm which gives a diversified key as an output to be stored in the card.

Due to the diversification process, the entropy of the tag's key decreases from 64

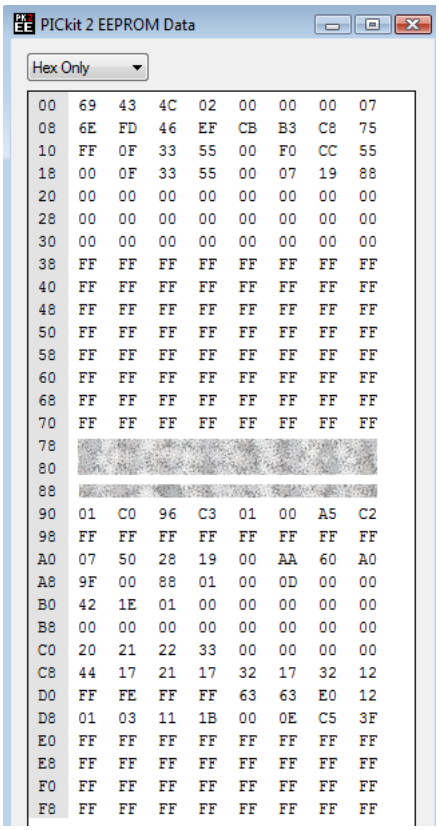

Figure 2–10: Memory read by PICkit2, the 16 byte 3DES data encryption key and the 8 byte authentication key are grayed out [60].

bits (of master key) to 56 bits (after single DES) minus the 2.2 bits of entropy that is showed by the authors to be lost during the diversification algorithm. The authors then demonstrate that the diversification algorithm consists of a hash function which is *neither one-way, nor collision resistant*. They built the reverse of the hash function and showed that recovering the iClass Master key is not harder than a chosen plaintext attack on single DES.

Two questions can be raised here:

- Why did the designers choose single DES over 3DES, while DES was already broken in 1997 [50] and HID readers are capable of computing 3DES?
- Why did the designers use their own proprietary hash function in the key diversification process, while they could use a more studied hash function like SHA-1 or MD5?

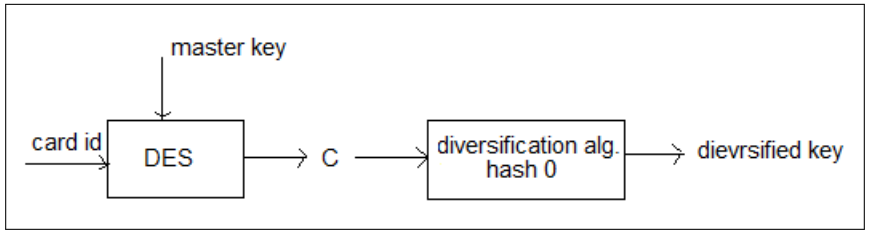

Figure 2–11: key diversification process

They started the reverse engineering process by analyzing many update card key messages from the reader. As a matter of fact, *iClass cards do not have a pseudo random number generator implemented on them*. Therefore, after any successful authentication, the reader sends an update message for block 2 of the tag to prevent replay attacks. This message updates the 32-bit card challenge c_c to be used in the future authentications. Unfortunately, there are no mechanisms in the card to ensure

the update with a new value.

The details of the process were significantly depending on the specification of the iClass cards and readers. The vulnerabilities that led to this attack are briefly presented:

- Flaws in the design of the internal hash function, which was not one-way nor collision resistant
- Absence of a pseudo random number generator on the card side
- No update verification by the card
- Use of DES algorithm in key diversification

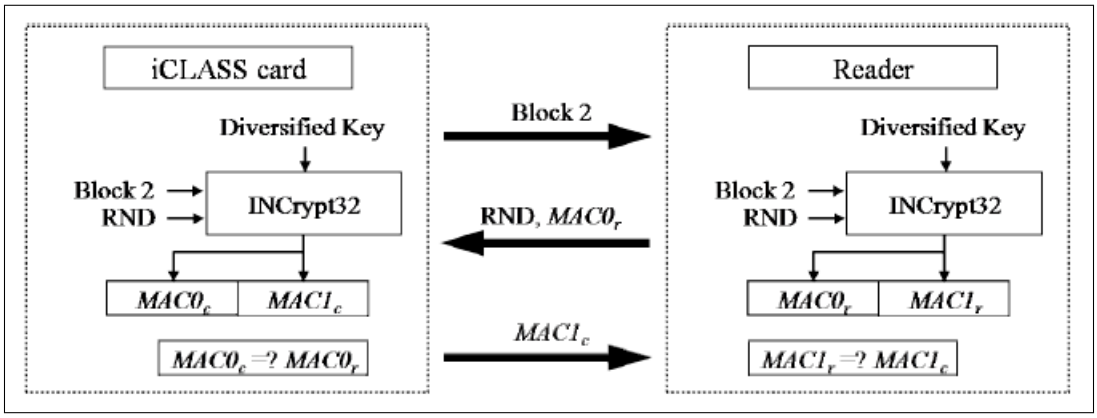

Figure 2–12: Authentication protocol [19]

The third attack was a full disclosure of the InCrypt32 proprietary algorithm. Figure 2-12 shows a description of the authentication process in iClass based on InCrypt32 algorithm [19]. As it is shown in the picture, the tag starts the authentication by sending its c_c (block 2) to the reader. The reader then uses c_c and a pseudo-random number plus the diversified key as inputs to INCrypt32 and generates two 4-byte signatures MAC0_c and MAC1_c. Reader sends the random number

along with $MAC0_c$ as a challenge to the tag. The tag calculates both $MAC0_c$ and $MAC1_c$ on his own and checks if the challenge sent by the reader matches the values it has calculated, if so, the reader is authenticated. The tag sends $MAC1_c$ to the reader to authenticate itself. The reader checks the received value with the one it has calculated and accepts the tag's authentication if they are equal. Kim et al. managed to discover the details of INCrypt32 algorithm through the reverse engineering of the hardware, a very similar approach that was used in the attack against Mifare Classic.

When the specifications of the INCrypt32 were revealed, the authors showed that this algorithms is vulnerable to chosen plaintext attacks. If the attacker is allowed to request MAC for arbitrary messages, then the secret key can be recovered within 2^{18} MAC queries, however if the length of the messages are limited to the specified value, then the required number of MAC queries grows to 2^{42}.

2.6 A Survey of Proposed Authentication Algorithms

In this section, some of the more important authentication algorithms along with their vulnerabilities and their strengths will be discussed.

Proposed authentication algorithms for low cost, passive RFID tags, can be categorized into two groups:

- Basic operation-based authentication
- Symmetric key authentication

2.6.1 Basic Operation-Based Authentication Algorithms

The group of basic operation-based authentication algorithms are also called Ultra Lightweight Mutual Authentication Protocols or UMAP. This category of

algorithms was initially introduced in Pedro Peris-Lopez's PhD thesis [43] with M2AP (Minimalist Mutual Authentication Protocol [12]). It was followed by EMAP (Efficient Mutual Authentication Protocol [11]) and LMAP (Lightweight Mutual Authentication Protocol [13]) by the same researcher.

Later, Prof. Chien from National Chi Nan University proposed his authentication algorithm named SASI, which was designed to be resilient against the vulnerabilities found in LMAP, EMAP and M2AP [20], [29], [30], but the SASI protocol had its own vulnerabilities that will be discussed in the following sections.

Pedro Peris-Lopez, then, proposed his new protocol based on the already defined protocols (SASI, LMAP) named Gossamer [18].

Among all, I have chosen LMAP, SASI and Gossamer to discuss in this chapter, mainly due to the larger number of articles devoted to their security analysis.

LMAP Protocol

LMAP stands for lightweight mutual authentication protocol. It is based on index pseudonyms (IDS) in which an IDS of 96 bits length is the index of a table showing all the information related to the tag in the backend system. Each tag has also a key which is composed of four parts of 96 bits each ($K = K1||K2||K3||K4$). So far, for the key and the IDS, $5 \times 96 = 480$ bits rewritable memory is required. Also, 96 bits of ROM memory are needed to store a static identification number (ID).

In the identification phase, the reader sends a "hello" message to the tag and the tag responds with its current IDS. With this IDS only an authorized reader will be able to have access to the secret key of the tag ($K = K1||K2||K3||K4$) (Figure 2-13).

The mutual authentication phase starts with the authentication of the reader. The

reader generates two random numbers $n1$ and $n2$. With $n1$, $K1$ and $K2$ it generates the submessages A and B. With $n2$ and $K3$ it generates submessage C. The reader is authenticated in this phase by the fact that it managed to access the database to $K1$, $K2$ and $K3$ (Figure 2-13).

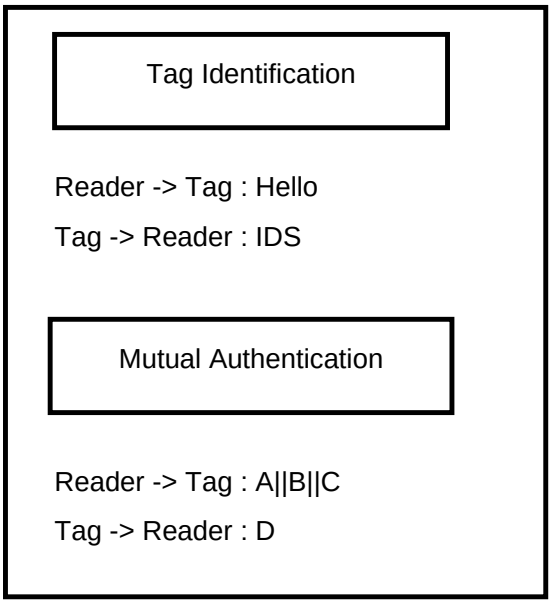

Figure 2–13: LMAP's identification and authentication

In the next step, tag should prove itself to the reader. It will first extract $n1$ and $n2$ from submessages A, B and C. These random numbers will be used for the update phase of the IDS. The tag generates the message D to authenticate itself and transmit its static identifier securely.

$$A = IDS^{n}_{tag(i)} \oplus K1^{n}_{tag(i)} \oplus n_1$$

$$B = (IDS^{n}_{tag(i)} \vee K2^{n}_{tag(i)}) + n_1$$

$$C = IDS^n_{tag(i)} + K3^n_{tag(i)} + n_2$$

$$D = IDS^n_{tag(i)} + ID_{tag(i)} \oplus n_1 \oplus n_2$$

Where $IDS^n_{tag(i)}$ is the IDS used for the n^{th} authentication with $tag(i)$. Also $K1^n_{tag(i)}$, $K2^n_{tag(i)}$ and $K3^n_{tag(i)}$ are the keys used in the n^{th} authentication with $tag(i)$.
After the authentication phase, the tag updates its IDS and $K1$ through $K4$, for the next authentication. The rules for these updates are as follow (Note that there is at least one random number n_1/n_2 in XOR with one of the $K1$ through $K4$ in each update rule:

$$IDS^{n+1}_{tag(i)} = (IDS^n_{tag(i)} + (n_2 \oplus K3^n_{tag(i)})) \oplus ID_{tag(i)}$$

$$K1^{n+1}_{tag(i)} = K1^n_{tag(i)} \oplus n_2 \oplus (K3^n_{tag(i)} + ID_{tag(i)})$$

$$K2^{n+1}_{tag(i)} = K2^n_{tag(i)} \oplus n_2 \oplus (K4^n_{tag(i)} + ID_{tag(i)})$$

$$K3^{n+1}_{tag(i)} = (K3^n_{tag(i)} \oplus n_1) + (K1^n_{tag(i)} \oplus ID_{tag(i)})$$

$$K4^{n+1}_{tag(i)} = (K4^n_{tag(i)} \oplus n_1) + (K1^n_{tag(i)} \oplus ID_{tag(i)})$$

No explanation about why these specific update rules have been chosen was provided.
The advantages of this protocol include:

- The designers of this protocol have used only simple operations such as bitwise XOR ($\oplus$), bitwise OR ($\vee$), bitwise AND ($\wedge$) and addition mod 2^m ($+$).
- The pseudo-random numbers generated by the reader in XOR with private keys are used as a source of entropy on the tag's side to avoid replay attacks.

This is to address the fact that it is hard to generate pseudo-random bits on the tag.

- A pseudo-random number (IDS) is sent every time for the identification phase which helps preserving the anonymity of the tag. This number looks random, due to the update rule, which consists of a pseudo-random number being XORed with part of the secret key and the ID of the tag.
- Memory and communication overheads are relatively low in this protocol. The communication overhead is only four messages (including the identification phase). The length of the messages differ, but the longest message is $3 \times 96 = 288$ bits.
- $K1$ through $K4$ along with the static ID of the tag are the secret information. 480 bits is in principle not vulnerable against brute force attacks with our current computational capacity.

However three main issues can be discussed regarding the LMAP protocol:

- LMAP is based on simple operations like bitwise AND, XOR, OR and sum mod 2^m. But all of these operations are triangular functions [20], which means that information does not propagate well from left to right. In other words, the bit in position i in the output only depends on bits $j : 1...i$ of the input.
- The use of bitwise AND and OR operations is a weakness in the LMAP protocol. The reason is that when a bitwise AND (OR) is computed over random bit streams the probability that the output is one (zero) is $\frac{3}{4}$, which is strongly biased and can help designing many passive attacks against this protocol.

- a simple attack that blocks the last message from the tag to the reader can de-synchronize the reader and the tag forever. Because the tag will update its keys after the last message but the reader considers the communication has halted and did not update.

SASI Protocol

SASI stands for Strong Authentication Strong Integrity and was proposed by Prof. Chien from Chi Nan National University in 2007. In this protocol, each tag has a static ID and a pseudonym IDS and two keys $K1$ and $K2$. The keys and the IDS are shared between the reader and the tag. The length of each of the ID, IDS, $K1$ and $K2$ is 96 bits.

Like LMAP this protocol consists of three phases: tag identification, mutual authentication and key updating phase. In this protocol though, each tag actually keeps two entries of (IDS, $K1$, and $K2$), one for the old values and the other for the potential next values. This mechanism has been applied to avoid the de-synchronization problem that exists in LMAP.

The operations in the SASI protocol are the same as LMAP except for a rotation operation. Rotation may be performed in several ways but restricted to the specifications of the protocol, it is not clearly specified which rotation method is used. Chien clarified later [17] that $Rot(x, y)$ is:

- $ROT(A, B) = A \ll wht(B)$ where $wht(B)$ stands for Hamming weight of vector B.

This is probably not the best security decision since the distribution of the rotated position is far from uniform [46]. This distribution is actually:

$$prob\{wht(B) = k\} = \frac{c(96, k)}{2^{96}}$$

The identification and the authentication of the protocol starts with the reader sending a "hello" message to the tag. The tag responds with its potential next IDS. The reader tries to find a match with the IDS in its database, if reader could find the entry in the database, it would start the authentication phase, otherwise it would query the tag again by sending another "hello" message. This time the tag responds with its old IDS. At the end of the mutual authentication process, the used entry will be stored in $(IDS_{old}||K1_{old}||K2_{old})$ and the updated values will be stored in $(IDS_{next}||K1_{next}||K2_{next})$.

In the mutual authentication phase, the reader uses the matched values in the database and two random number $n1$ and $n2$ to compute the values A, B and C. The reader will send $A||B||C$ to the tag and the tag extracts $n1$ from A, $n2$ from B and computes $\bar{K1}$ and $\bar{K2}$ and verifies the value of C. The fact that the reader could have access to $K1$ and $K2$ shows that it is a legitimate reader. The tag needs to compute the message D and send it to the reader to prove itself. After the mutual authentication phase, both reader and tag need to update their values, according to the equations in Figure 2-14.

This protocol is also very light and uses only the basic operations that can be implemented within the limited computational capacity of a low cost RFID tag.

It consists of only 4 messages for identification and mutual authentication, and very

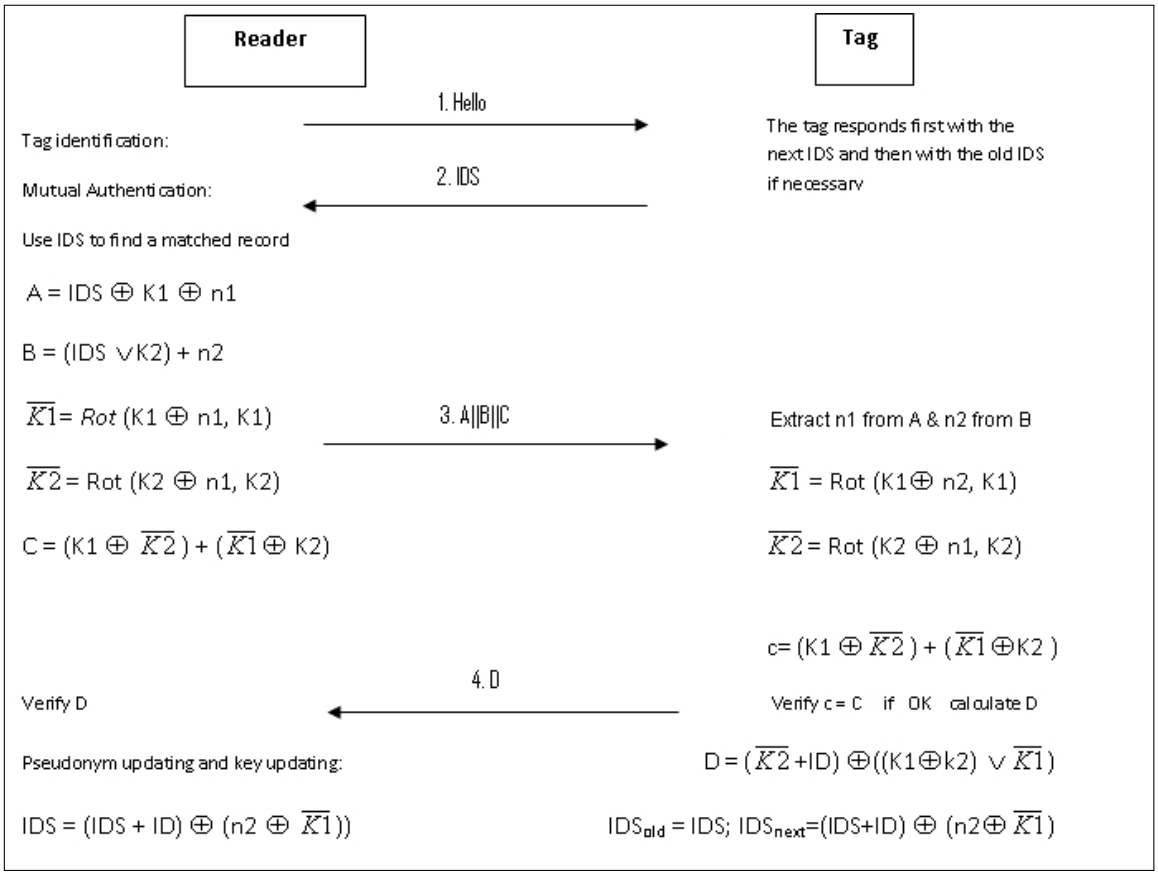

Figure 2–14: identification and mutual authentication in SASI protocol

small rewritable memory for the tag. Moreover, it preserves the anonymity of the tag by using pseudorandom IDS every time.

However, security analysis of the protocol by [17], [43] showed that:

- Although ROT operation is included in the protocol as a non triangular function, when the number of rotations is 0 or 96 we are left with the same set of all triangular functions in the protocol, the same vulnerability as in LMAP. This was the base of an attack to discover the ID of the tag by [43].
- The IDS update is dependent on n_2 and $\bar{k_1}$ which is again a function of n_2. This is used in [43]'s attacks and can be used in other attacks as well, due to the poor statistical properties it generates.
- It as shown in [17] that this protocol is still vulnerable against de-synchronization attack that can disable the tag from future authentications. They showed that by interrupting the communication, the attacker can replay the previous recorded communication with the tag (due to the lack of random number generator on the tag's side) and update the secret information on the tag's side without doing the same on the reader side, which causes a de-synchronization between the reader and the tag.

Gossamer Protocol

The other important protocol among the ultra lightweight authentication protocols is Gossamer. This protocol was proposed by Lopez et al in 2008 [18]. Gossamer was also designed with the purpose of using only basic operations. It uses all the functions in SASI and beside them, it uses a MixBits function, which is defined to evolve compositions of extremely light operands, by means of genetic programming,

in order to obtain highly non-linear functions. The MixBits function is shown in Figure 2-15.

```
Z = MixBits (X,Y)

--------------------

Z=X;

For(i=0;  i<32; i++){

      Z= (Z>>1)+Z+Z+Y;

}
```

Figure 2–15: MixBits function in Gossamer Protocol

The identification of the tag is the same as SASI protocol, a "hello" message from the reader is sent, the tag responds with its potential next IDS, if the reader can find a matched entry in its database the authentication process starts otherwise, the reader would ask for IDS again by sending another "hello" message, and this time, tag responds with its old IDS.

The mutual authentication process is also the same as SASI protocol, with different formulas for message A,B, C and D and also different update rules. Therefore, when the reader finds a matched entry for the tag in the database, it starts to calculate A, B and C according to the following formulas:

$$A = ROT((ROT(IDS + k_1, \pi + n_1, k_2) + k_1, k_1)$$

$$B = ROT((ROT(IDS + k_2 + \pi + n_2, k_1) + k_2, k_2)$$

$$n_3 = MIXBITS(n_1, n_2)$$

$$k_1^* = ROT((ROT(n_2 + k_1 + \pi + n_3, n_2) + k_2 \oplus n_3, n_1) \oplus n_3$$

$$k_2^* = ROT((ROT(n_1 + k_2 + \pi + n_3, n_1) + k_1 + n_3, n_2) + n_3$$

$$n_1^{'} = MIXBITS(n_3, n_2)$$

$$C = ROT((ROT(n_3 + k_1^* + \pi + n_1^{'}, n_3) + k_1^* \oplus n_1^{'}, n_2) \oplus n_1^{'}$$

In which $\pi = 0x3243F6A$.

The message $A||B||C$ will be sent to the tag. Tag extracts n_1 from A, and n_2 from B, and calculates $C^{'}$ according to the following equations and checks if $C = C^{'}$. if so, reader is authenticated and message D will be sent back to the reader.

$$n_3 = MIXBITS(n_1, n_2)$$

$$k_1^* = ROT((ROT(n_2 + k_1 + \pi + n_3, n_2) + k_2 \oplus n_3, n_1) \oplus n_3$$

$$k_2^* = ROT((ROT(n_! + k_2 + \pi + n_3, n_1) + k_1 + n_3, n_2) + k_1 + n_3, n_2) + n_3$$

$$n_1' = MIXBITS(n_3, n_2)$$

$$C' = ROT((ROT(n_3 + k_1^* + \pi + n_1', n_3) + k_2^* \oplus n_1', n_2) \oplus n_1'$$

if $C = C'$

$$D = ROT((ROT(n_2 + k_2^* + ID + n_1', n_2) + k_1^* + n_1', n_3) + n_1'$$

Reader calculates D', and checks if $D = D'$, and if so the tag is authenticated and both reader and tag will update their secret information.

$$D' = ROT((ROT(n_2 + k_2^* + ID + n_1', n_2) + k_1^* + n_1', n_3) + n_1'$$

Updating rules for the tag and the reader are as follows:

The tag updates according to the following rules:

$$n_2' = MIXBITS(n_1', n_3)$$

$$IDS^{old} = IDS$$

$$IDS^{next} = ROT((ROT(n_1' + k_1^* + IDS + n_2', n_1') + k_2^* \oplus n_2', n_3) \oplus n_2'$$

$$k_1^{old} = k_1$$

$$k_1^{next} = ROT((ROT(n_3 + k_2^* + \pi + n_2', n_3) + k_1^* + n_2', n_1') + n_2'$$

$$k_2^{old} = k_2$$

$$k_2^{next} = ROT((ROT(IDS^{next} + k_2^* + \pi + k_1^{next}, IDS^{nxt}) + k_1^* + k_1^{next}, n_2') + k_1^{next}$$

and the back-end system updates according to the following rules:

$$n_2' = MIXBITS(n_1', n_3)$$

$$IDS = ROT((ROT(n_1' + k_1^* + IDS + n_2', n_1') + k_2^* \oplus n_2', n_3) \oplus n_2'$$

$$k_1 = ROT((ROT(n_3 + k_2^* + \pi + n_2', n_3) + k_1^* + n_2', n_2'$$

$$k_2 = ROT((ROT(IDS + k_2^* + \pi + k_1, IDS) + k_1^* + k_1, n_2') + k_1$$

As of the other light weight protocols, Gossamer uses only bitwise operations which makes it very low cost in terms of the computational capacity. Mutual authentication is (non-provably) provided due to the fact that only a legitimate reader can compute message $A||B||C$ with access to (k_1, k_2) and only a legitimate tag can extract n_1 and n_2 from $A||B||C$ and compute D. Moreover, messages C and D including the secret

values $(n_3, n_1', k_1^*, k_2^*)$ and nonce (n_1, n_2) allow data integrity to be checked.
Although Gossamer could improve some of the weaknesses in SASI protocol, the same de-synchronization attacks designed against SASI can still be launched against Gossamer as well to disable the tag from future authentications.

2.6.2 Symmetric-key based Authentication Algorithms

The second group of authentication protocols are based on simple symmetric-key cryptographic algorithms. In this group, the proposed algorithms try to apply simple cryptographic functions to the world of RFID systems. Among them T2MAP and HB family protocols including HB, HB+ and HB++ are of relative notoriety and will be discussed in this subsection.

T2MAP

T2MAP stands for Two-Message Mutual Authentication Protocol and was proposed by Selma Boumerdassi et al. in 2006 [21]. The protocol uses pre-computed values stored in arrays of two columns on both reader and tag sides. Some initial pairs of ID and the corresponding keys of the tag are saved in the first and the second column of the array, respectively. The tag and the reader will update the values in this array regularly to make sure that the same values will not be used twice. This process adds the necessity of keeping the values on the tag side equal to the values on the reader side, because these values are going to be used as shared secrets between the reader and the tag during the authentication process.
Communication starts by the reader sending two ID from the array to the tag. These ID are chosen randomly and are not encrypted. The designers of the protocol have decided to choose two ID at this step, because a single ID was not enough to provide

the desired level of security for the designers. More specifically, with an array of size n, the probability that a specific ID is chosen is $O(\frac{1}{n})$ while in the case of two ID this probability is $O(\frac{1}{n^2})$.

Tag verifies that these two ID exist in its array and if so, it will extract the two corresponding keys, applies a function f to them and sends the result back to the reader (Note that the elements in the second column are never sent in clear.). If the two ID or one of them does not exist in the array, the tag halts the communication. The reader checks if the result received from the tag matches the result of applying the same function on the corresponding keys on its own array. If yes, the tag is authenticated, if not, reader will not trust the tag for the rest of the communication. An illustration of the protocol is shown in Figure 2-16.

Applying this protocol in a "trustworthy channel" between reader and tag seems reasonable. "Trustworthy channel" here means a channel through which the messages do not get lost and do not get corrupted (although the channel can still be eavesdropped). To be applicable for RFID systems for which the channel is not trustworthy, the designers have changed the protocol in a way that the reader stores the current values of the array along with the old values in order to avoid de-synchronization with the tag. Therefore, if any de-synchronization happens between the tag and the reader as a result of a malicious frequency jamming or the message loss/corrupt reader and tag will be able to recover. The reader will inform the tag, regularly enough to update its array according to a pre-shared update rule that consists of only basic operations such as bitwise AND and bitwise XOR.

T2MAP does not require large bandwidth because it only takes two messages to

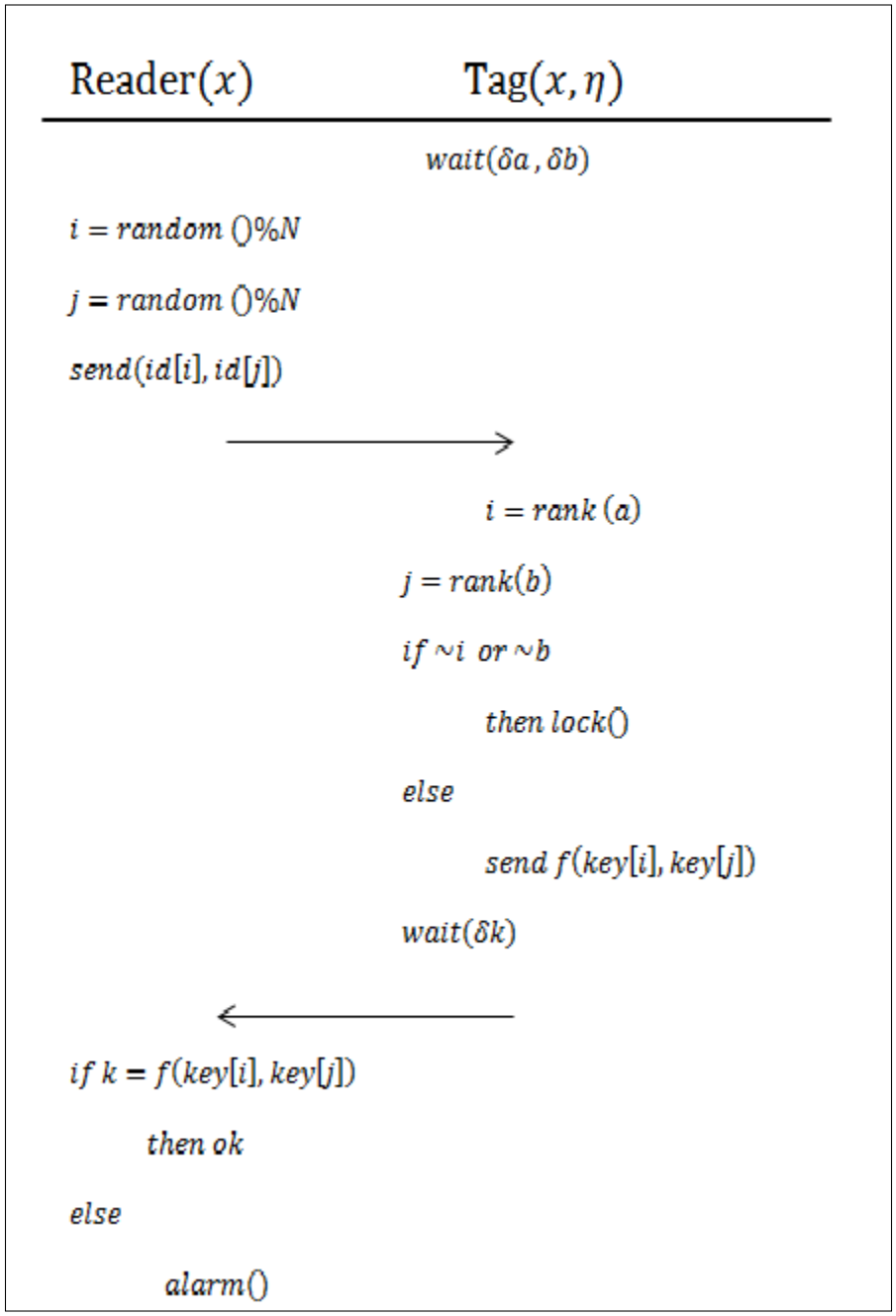

Figure 2–16: T2MAP identification and authentication

mutually authenticate the parties in the communication. Moreover, no information related to the array is transmitted between the tag and the reader. The idea of T2MAP along with a smart set of update rules and an optimum size of the array may provide a very strong authentication protocol for RFID tags.

HB family protocols

HB (Hopper-Blum) protocol was proposed by Hopper and Blum in 2001 [22]. The security of the proposed protocol against passive attacks is rooted in the *learning parity with noise problem*, which is believed to be NP-hard.

The following variables are involved in the protocol:

- a, b: k-bit random vectors.
- x : k-bit secret key vector, pre-shared between the reader and the tag.
- υ: noise bit (which is 1 with probability $\eta \in [0, \frac{1}{2}]$)

In each round of the protocol, the reader randomly generates a and sends it to the tag. The tag calculates $z = a \cdot x \oplus \upsilon$ and returns z to the reader. The reader then checks if $a \cdot x \approx z$.

The protocol is operated r times and the tag is authenticated if the check on the reader side fails at most $\eta \cdot r$ times.

HB protocol is believed to be secure against passive attacks but an active attacker can retrieve x by sending multiple values of a to the tag and analyze the result of the authentication (success/failure) to compute the secret x.

Therefore, HB+ was introduced by Juels and Weis in 2006 [48], which is a modified version of HB to be resilient against the active attack described above. The differences between HB and HB+ are threefold:

- another k-bit secret key y is introduced to be shared between the reader and the tag
- the tag and not the reader initiates the communication in HB+ with a random vector b
- z is computed as $z = a \cdot x \oplus b \cdot y \oplus v$

Therefore, the tag initiates the authentication by sending a random vector b to the reader. The reader also sends back a random vector a to the tag. The tag computes $z = a \cdot x \oplus b \cdot y \oplus v$ and sends z back to the reader. The reader checks if $a \cdot x \oplus b \cdot y \approx z$. This round will be repeated r times and the number of failed attempts should not exceed a desired threshold. Notice that the protocol is assuming the tag to be capable of using a random number generator (or a pseudo-random number generator), which was not the case in the previous version of this protocol.

Schematic of the HB protocol and HB+ protocol are shown in Figure 2-16 and Figure 2-18.

However, in 2005 Gilbert et al. [52] showed that HB+ is vulnerable against a simple

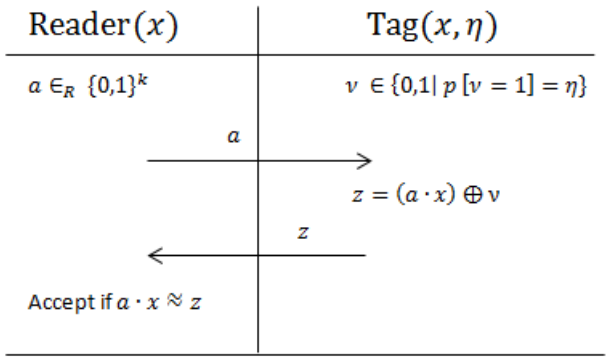

Figure 2–17: HB protocol

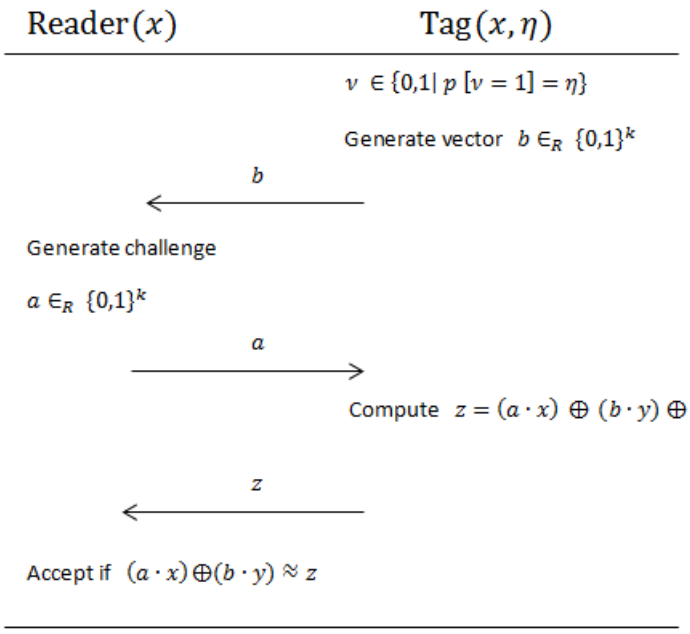

Figure 2–18: HB+ protocol

man-in-the-middle attack. The adversary can manipulate the challenge sent from a legitimate reader to a legitimate tag during the authentication process, he should also be able to recognize if the authentication has failed or is passed. More specifically, the reader sends a to the tag, but the adversary changes the challenge by XORing it with a constant k-bit vector on all r rounds of the authentication process. If the authentication succeeds, with a noticeable probability, then $\delta \cdot x = 0$, otherwise $\delta \cdot x = 1$. Using carefully chosen values for δ, the adversary can determine various bits of x. In the worst case, the adversary will be able to recover all k bits of the secret x by repeating this process k times.

Protocol	Year	Attack	Year
HB	2001	active	2001
HB+	2005	GRS	2006
HB++	2006	GRS	2008
HB-MP	2007	passive	2008
HB*	2007	GRS	2008
HB-MP+	2008	passive	2008

Table 2–1: HB family authentication algorithms and the attacks against them. GRS stands for the attacks designed by Gilbert, Robshaw and Sibert

In response to the vulnerabilities in HB+, Bringer et al. proposed HB++ in 2006 [56], which was introduced to have the strengths of the previous versions of this protocol and have resiliency against the man-in-the-middle attack; however, HB++ and even the version after it [57] were also some ad-hoc attempts to build a secure authentication algorithms for RFID systems, for which attacks were designed not more than a year after. Table 2-1 shows a brief summary of the HB family authentication algorithms and the attacks against them [36].

2.6.3 Summary and Conclusions

The past two sections were the parallel attempts of industry and academia to provide some level of security and privacy for RFID systems (Refer to Table 2-2 and Table 2-3 for a summary).

On one hand, RFID manufacturers try to get security through obscurity, which is deemed a failure like the previous examples of TI DST, Mifare Classic and HID iClass. On the other hand, authentication/encyrption schemes proposed by academia can not provide a desirable level of security/privacy for these systems as attacks have

been designed against them not more than a year after they were proposed.

Victim System	Designer/Year	Type	Cost	Success at	Main vulnerability
IT DST	Bono/2005	Brute Force	a few 100$	cloning	Short key
Mifare	Nohl/2008	Hardware rever.	$\leq$ \$1000	key recov.	Short PRNG
Mifare	Gans/2008	Protocol rever.	not much	key recov.	Short PRNG
HID iClass	Meriac/2010	Hardware analy.	a few 100$	cloning	Mem. config.
HID iClass	Garcia/2011	Cryptanalysis	not much	key recov.	Weak Hash Func.
HID iClass	Kim/2011	hardware rever.	$\leq$ \$1000	replay att.	Weak Crypt. Func.

Table 2–2: Previous Attacks Summarized

Proposed Sol.	Year	Type	Vulnerabilities	Year of Attack
LMAP	2006	UMAP	T-Func, bitwise op., de-synch.	2006
SASI	2007	UMAP	T-Func, weak Rot, de-synch	2007
Gossamer	2008	UMAP	de-synch	2008
T2MAP	2006	Symm. key	de-synch	2007
HB	2001	Symm. key	passive	2001
HB+	2005	Symm. key	GRS	2006
HB++	2006	Symm. key	GRS	2008

Table 2–3: Proposed Solutions Summarized

CHAPTER 3
Security Analysis of OPUS Cards

3.1 Introduction

OPUS is an RFID card used in the Island of Montreal for access control to the public transportation. Beside the STM (Société de transport de Montréal), other agencies such as RTL (Reséau de transport de Longueuil) and STL (Société de transport de Laval) have also transferred to these smart cards for authentication and access control to public transport.

Figure 3-1 shows the front and the back of an OPUS card. On the front, there is

Figure 3–1: An OPUS Card, front on the left and back on the right

a small chip which is responsible for computations and communications of the card with the reader. On the back, there is a unique 10 digit serial number, which is the UID of the card to be used in anticollision and authentication.

According to a study in 2006, 21.4 % of the people in Montreal use public transportation for their daily commute [61]; therefore, Montreal is the second city in Canada

in terms of the public transport use, after Toronto with 22.2%. This number shows the importance of the public transport in the citizen's life, and consequently the responsibility that the agencies such as STM hold for the security and the privacy of the card holders.

STM has started using OPUS cards in 2008 and has improved and broaden its service during the past four years [62].

OPUS cards are the product of Oberthur Technologies[1] and are fully compatible with Calypso which is a standard for smart cards in transportation services. Calypso was developed by a group of European partners from Belgium, Portugal, Germany and France. It consists of a set of technical specifications including a fast and secure contactless transaction between a terminal (reader) and a portable device (tag), as described in the Calypso Handbook [32].

52 million Calypso cards and 260,000 Calypso readers had been sold by the beginning of 2010, which makes it the largest microprocessor-based smart card and technology ticketing system. Also, over 900,000 OPUS cards have been delivered to Quebec transportation system agencies since the service has been launched [32].

According to the Calypso standard, the complete transaction is performed in less than 200 ms, including anticollision, mutual authentication and exchange of encrypted messages. It is also claimed that even if the card is withdrawn too quickly or if the communication is broken, the protocol can preserve the integrity of the data

[1] http://www.oberthur.com A.T: 7/4/2012

written on the card.

Each card has three secret keys of **16 bytes** each:

- **issuer key**: used to modify personal data
- **load key**: used to modify reload data
- **debit key**: used to modify validation data

Any of the keys can be used to authenticate the content of the card.

The card's secret keys are derived from SAM master keys and the Calypso serial number of the card. In the literature of Calypso, the SAM (Secure Application Module) is a smart card that authenticates the card (Portable Object-PO), the reader (the terminal) and the data in exchange. Historically, SAM was a smart card but now it can be a piece of hardware plugged into a server (Figure 3-2). The security specifications of Calypso (including SAM specifications) are available only after signing a non-disclosure agreement and payment of $1000.

Following is a set of privacy weaknesses shown about OPUS cards through a set of

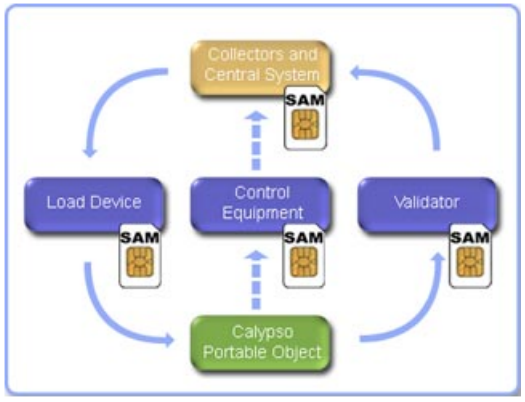

Figure 3–2: Calypso ticketing systems components and their security module

experiments. This chapter is followed by results and discussion about the possible attack scenarios against OPUS cards.

3.2 Tools and Methodology

In order to study the security and the privacy of OPUS cards, several traces of communication between a reader and a card were recorded. Different readers (metro and bus stations) as well as different cards (monthly pass, daily pass and token based) were used to gather the data.

3.2.1 Tools for recording the communications

Communications can be recorded by any device capable of receiving radio frequency signals. The gathered RF signals should be processed to retrieve the message transmitted between the two parties.

The following two devices were used in this project:

- the Universal Software Radio Peripheral (USRP) which is a high speed USB board and is controlled via open source software called GNU radio for the signal processing phase. A simple USRP can be bought online for around $700 [39]. RF signals that are sniffed by the USRP can be processed with a higher level language such as MATLAB for decoding and post processing phases. Figure 3-3 shows a USRP device.
- the Proxmark3 (Figure 3-4) board designed specifically to work with both high frequency and low frequency RFID signals. It is composed of an antenna, an Analog to Digital Convertor (ADC), a Field Programmer Gate Array (FPGA), a micro controller, a micro processor (ARM) and a USB port. The Proxmark3

is designed to snoop, listen and emulate every signal from different RFID standards, such as 14443a and 14443b. Conversation between the reader and the tag can be printed in hexadecimal on the screen or in a file on the computer connected to it via the USB port. The Proxmark3 is available for around $400 and can be ordered online [40].

Figure 3–3: A USRP device

Figure 3–4: A Proxmark3 device

The communication between a card and a reader can be sniffed using any of the two devices. Moreover, since both devices are capable of transmitting RF signals at

appropriate frequencies, it is possible to emulate both the card and the reader.

Marco Bottino, a former student in the lab, had recorded several traces of communication between a reader and an OPUS card using a USRP, in 2008[26]. He started developing several MATLAB files to process the sniffed communications, and to emulate an OPUS reader, but the development of the tools was not complete. When we started in 2010, we could continue working on the development of the same set of files, or on the adaptation of the Proxmark3 to analyze OPUS cards[2] .

We chose to work with the Proxmark3 because:

- it was developed specifically to work with RFID systems and much of the job (signal processing and information display functions) was already implemented and was working perfectly.
- the Proxmark3 was much smaller and lighter than the USRP and it was more practical with our particular requirements. More specifically, we needed to sniff the communications within the metro or bus stations because OPUS readers were not available in the lab, and with the Proxmark3 we could hide our investigations to avoid raising suspicions.

In the Proxmark3, the FPGA receives the digital signal from the ADC and is responsible for the digital signal processing (DSP) computations, such as edge detection. The ARM Microcontroller handles the encoding/decoding the signal (Manchester, Miller, etc.) as well as more advanced functions. We worked on the ARM code to

[2] OPUS cards' contactless protocol is close to ISO 14443b with a few significant differences.

make it work with the OPUS card, as the functions implemented for ISO 14443b was helpful, but not enough to sniff the communications.

One other technical problem was that the Proxmark3 was powered through a USB connection which was not available in the stations. Therefore, a Duracell USB charger was used to power the Proxmark3 during sniffing. The data (a hexadecimal byte stream) was later transferred to the computer in the lab for further processing.

Through some preliminary experiments it was recognized that the first few exchanges were not random, meaning that they could not be part of the authentication process. We expected the communication to start with an anticollision process. The reader always started the process by sending the constant bit stream:

05 00 00 71 FF

which could be considered as a "hello" message. The OPUS card answered with the bit stream:

50 92 4E B9 46 00 00 00 00 00 71 71 85 B7

which was constant for the same card. We repeated the experiment with different cards and could figure out which bytes were representing the UID of the card.

We were aware of two facts about the public transit system:

- two hours validity for each ticket, and
- prohibition to use the same bus line with the same ticket twice

Therefore, we guessed that the first few exchanges after the anticollision process were verifying the last usages of the card. We started recording more sessions with varying

parameters like the card, the metro station, and the time of the day and we came up with a set of hypotheses about the meaning of the first few exchanges.

After the preliminary experiments we were quite sure which bytes of the recorded bit streams were representing the time and the date, but figuring out the way the time and the date were represented was another story which was only possible after precise experiments with the card at different dates and times. At the end we developed a java application (PenguinInterface.java in Appendix A) which would extract the information about the card's UID and the last usages of the card from the messages and represent it in human-readable format.

In the next section, the results of our experiments are explained which can be verified easily with the same set of tools and in any metro/bus station.

3.3 Results

We went to one of the metro stations to buy two OPUS cards. Surprisingly, we realized that the numbers printed on the back of the cards are not random but they are assigned just sequentially. The evidence is brought in Figure 3-5, which is the receipt of the two OPUS cards bought at the same time. If the UIDs were generated randomly, the probability of getting such a result would be of the order of 10^{-10} which is negligible.

Fact-1: The UID of the cards are generated sequentially.

By sniffing the communication between the cards and the readers, we realized the ID printed on the back of the card is queried by the reader at the beginning of the communication.

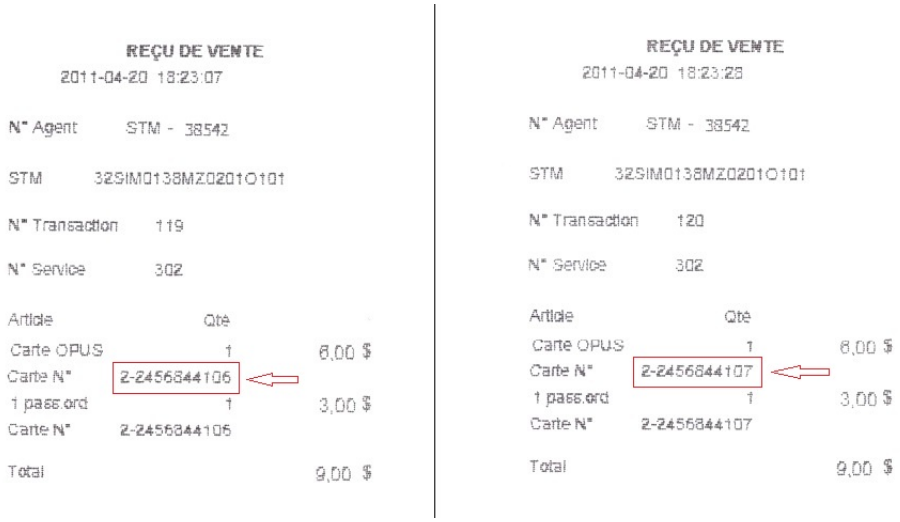

Figure 3–5: Receipts of the two OPUS cards bought at the same time.

The reader starts the communication by sending the constant bit stream

05 00 00 71 FF

and the card answers with the bit stream

50 **92 4E B9 46** 00 00 00 00 00 71 71 85 B7

in which **92 4E B9 46** is the static UID of the card in hexadecimal referring to **2-2454632774** printed on the back.

The UID is sent in clear which makes it possible for an adversary to query a card without the owner's consent, by placing a Proxmark3 in the vicinity of it.

> **Fact-2:** The static UID of the cards are printed on the back of the cards and is sent in clear in the beginning of the communication.

After the first exchanges, the reader repeats the UID of the tag in the next query (Table 3-1) which makes eavesdropping much easier for the attacker. Since

msg number	Sender	Message in hex
1	Reader	05 00 00 71 FF
2	Card	50 **92 4E B9 46** 00 00 00 00 00 71 71 85 B7
3	Reader	1D **92 4E B9 46** 00 08 01 01 5B 4A
4	Card	01 F1 E1
5	Reader	0A 01 94 B2 01 44 1D 5D 9A
6	Card	0A 01 **51 3D 2B** 80 00 0C F0 00 **10 03 28 00 08** C7 A8 ...
7	Reader	0B 01 94 B2 02 44 1D EC EA
8	Card	0B 01 **51 36 AD** 00 00 0C F0 00 **10 03 28 00 10** C7 A8 ...
9	Reader	0A 01 94 B2 03 44 1D E5 2F
10	Card	0A 01 **51 36 50** 80 00 0F F8 00 00 **10 03 28 00 08** 00 00...

Table 3–1: Beginning of a communication session between an OPUS card and a reader

the reader-to-tag channel range is much larger than the tag-to-reader channel range, if the eavesdropper has missed the UID in the first response of the tag he gets another opportunity to sniff it in the reader's next challenge, but with a stronger signal this time.

After the anticollision process, the reader queries the card for the last three successful authentications of the card. The card answers with the time and the bus line/metro station of the last three usages.

The reader asks about the x^{th} last usage with the bit stream:

0A(0B) 01 94 B2 0***x*** 44 1D 5D 9A

If x is odd (1 or 3) bit stream starts with 0A, otherwise (x=2) it starts with 0B. The card answers to this query with the bit stream:

0A 01 **51 3D 2B** 80 00 0C F0 00 **10 03 28 00 08** C7

A8 9E 95 C2 00 00 00 00 00 00 00 00 00 00 00 90 00 A4 FE

51 3D 2B refers to the date and time and **10 03 28 00 08** refers to the station of the x^{th} usage of the card.

> **Fact-3:** The date, the time and the station of the last three usages of the card are sent in clear before the authentication.

From the 24 bits that represent the date/time, the first 14 bits are the number of days passed since 01/01/1997 and the last 10 bits are the time of the day: half the number of minutes passed since 00:00:00, i.e. the time is stored with a max precision of 2 min.

From the 40 bits that represent the station, the first 24 bits shows if the card has been used in a metro station or in a bus. **10 03 28** shows usage in metro while **10 00 00** shows usage is a bus. The next 16 bits shows more specifically the metro line or the bus number. The mapping in Table 3-2 has been deduced from several samples gathered.

The other mappings between the bus lines/metro stations and the bit streams can also be deduced with more experiments.

Line	Hexadecimal number
Metro Green Line	10 03 28 00 08
Metro Orange Line	10 03 28 00 10
Bus 80	10 00 00 01 D8
Bus 129	10 00 00 02 E8
Bus 535	10 00 00 05 60
Bus 165	10 00 00 03 78
Bus 144	10 00 00 03 30

Table 3–2: Mapping of bus-line/metro stations and the hexadecimal numbers

Recorded communications in 2008 by Marco Bottino, a former student in the lab,

shows that at the time only the last usage of the card had been queried by the reader[26]. For some reason, STM has decided to increase this information retrieval to the last three usages, since then.

A sample of the results converted to a human readable format by the PenguinInterface.java application is shown in Figure 3-6. In the first row and the first column, the UID of the tag is printed. The date, time and the station/line where the card has been used is deduced from the recorded communication.

Figure 3–6: Human-readable results from the communication between a card and a reader

3.4 Description of possible attack scenarios

The results of the experiments in the previous section can be the subject of several attacks against the card holders. These attacks can be based on the information leakage from the system or simply based on the way the system works. We will study the possible attack scenarios in this section.

3.4.1 Information leakage attacks

Information leakage can happen automatically and without the user's knowledge due to the contactless communications which is the main feature of RFID systems. There are three ways in which an adversary can launch an information leakage attack:

1. The unique serial number of the card can be stolen through sniffing, skimming or simply looking at the back of the card. Knowledge of this ID, can be used to track or even identify the card holder. This identification is based on the fact that the UID is associated with some other information about the card holder, such as when he has bought it, where he lives/works, or at what time does he commutes. This correlation statistically increases the possibility of identifying a person. Although this is probably not the only or even the best way of tracking or identifying a person, it is still a privacy violating feature and is not inevitable. Unfortunately the tracking and identification can be done at distance and in an automatic way and without the card holder's knowledge to create a customer profile for aimed advertisement, or track the card holder for any other reason.
2. The fact that the unique serial numbers are generated sequentially, can help the adversary to estimate the time when the OPUS card has been bought. We tried to do the same among the students in the lab and the information about the month/year of the purchase could be easily deduced from the gathered IDs. A more precise attack can reduce this time slot to a day or even less.
3. Information about the last three usages of the card (Bus-Line/Metro-Station) can be sniffed through the communication of the card with a legitimate reader. It can even be skimmed with a Proxmark3 in the vicinity of the card. This information can be used to deduce the time of the daily commutes of the card holder or about his home/work neighborhood.

3.4.2 Ticket fraud attacks

Through a man-in-the-middle attack, an adversary can recharge a card at a point of service, without letting the terminal know about that. This attack is based on the weak protocol design in Calypso systems. The communication protocol between a card and a reader is described in a publicly available document in [51]. According to this document, the signal *Signature Hi* is sent by the reader to inform the card that the transaction on the reader side is done. Upon receiving this signal, the card commits the transaction on its side and if successful, sends the signal *Signature Low* to the reader. The reader commits the data on its side when it receives *Signature Low*, and cancels the transaction in case the signal is not received.

Therefore, if the adversary manages to prevent the signal *Signature Low* from reaching the reader, he has managed to have the reader cancel all the previous transactions, while the card has been charged and the new number of tickets is stored in the memory.

One way of preventing the *Signature Low* signal from reaching the reader is having a Proxmark3 listen to the radio-frequency communication between the two parties. When *Signature Hi* is sent by the reader, the Proxmark3 starts sending random data to generate interference and hence preventing the reader from receiving the *Signature Low* from the card (Figure 3-8). Another way of preventing the signal *Signature Low* from reaching the reader is to use two Proxmark3 as the card and the reader simulators which relay all the messages, except for *Signature Low*, between the real parties (Figure 3-9). Relaying the data between the two parties adds a non-negligible delay which is not compatible with the strict timing requirements of the anti-collision

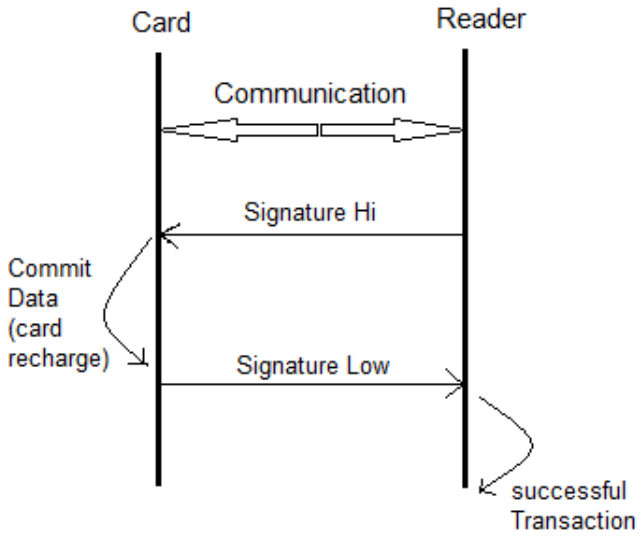

Figure 3–7: Communication protocol between a card and a reader for a successful card recharge

phase. For this reason, the Proxmark3 in the role of card simulator will be pre-loaded with the ID of the card to answer the anti-collision request of the reader directly from the memory.

In another type of man-in-the-middle attack the adversary can take advantage of the plain messages sent at the beginning of the communication. Every OPUS ticket is valid until 2 hours from the time it is first used, hence an adversary can change the messages regarding the last three usages of the card and use the same ticket for as long as he wants (Figure 3-10). The modifications in Figure 3-10 can be for or against the card holder. The adversary can modify the time/date in the message to convince the reader that an invalid ticket is still valid or a valid ticket is expired, which is a serious attack, either way.

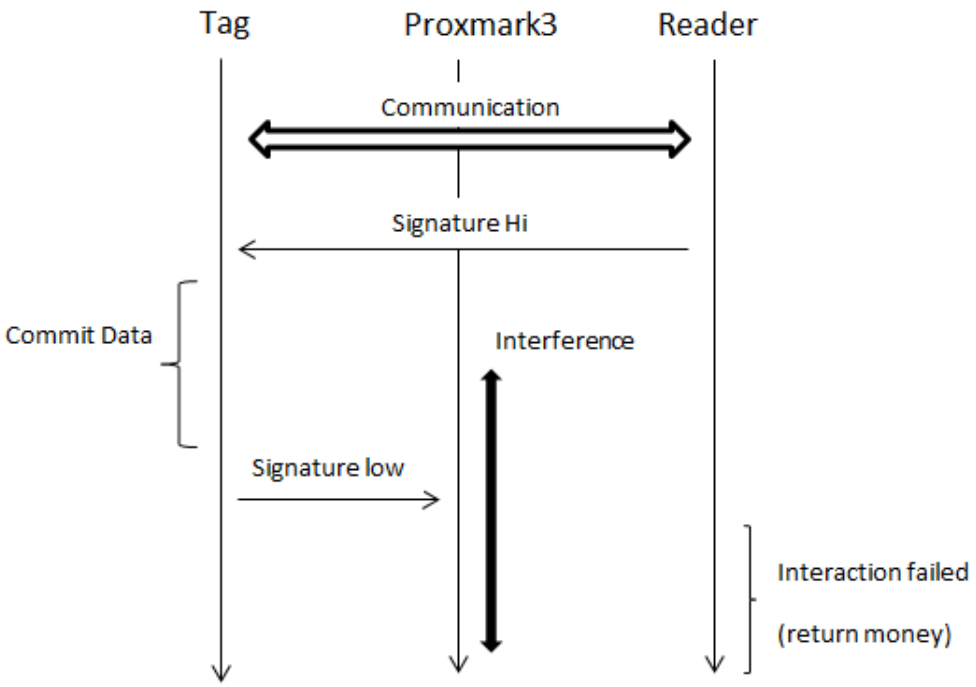

Figure 3–8: A man-in-the-middle attack to charge the card without payment

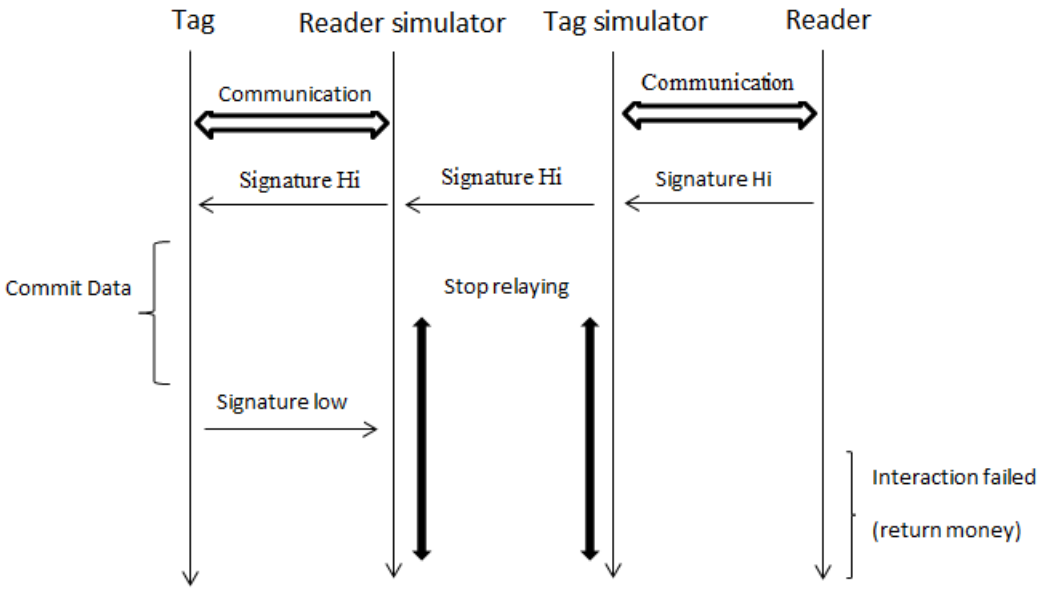

Figure 3–9: Another man-in-the-middle attack to charge the card without payment

The adversary requires a $400 Proxmark3, to complete any of the information leakage or ticket fraud attacks.

While these attacks are theoretically possible and achievable with the tools developed in the lab, we chose not to experiment them with real OPUS cards for legal reasons.

3.5 Security of the authentication protocol

To gather adequate authentication sessions for the study, we faced two problems a) we could not have access to the OPUS readers in the lab and hence we needed to record the communications in the metro/bus stations, and b) we had to record the sessions while attaching our Proxmark3 antenna to the OPUS card because the antenna was very weak and simulating the eavsdropping attacks from distance was

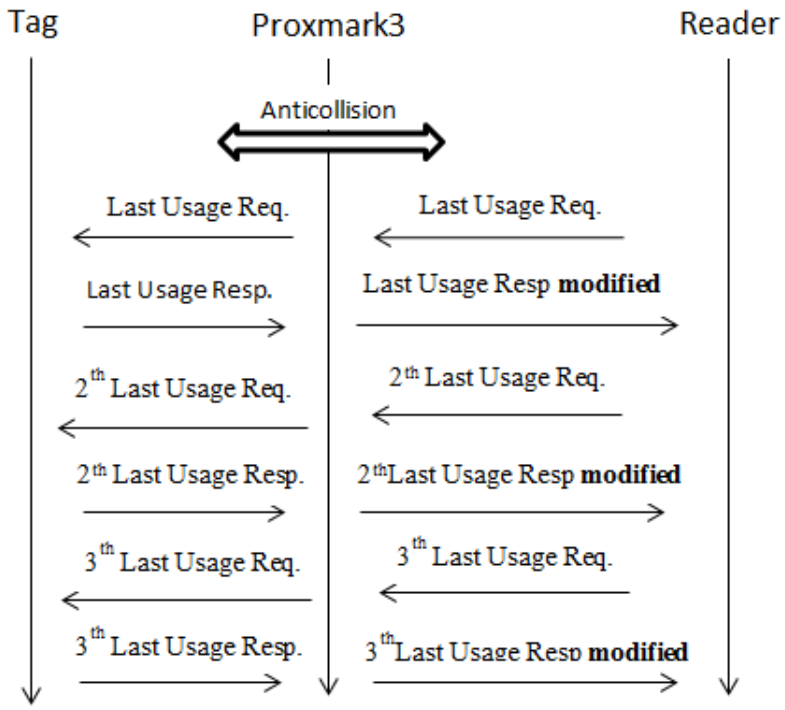

Figure 3–10: A man-in-the-middle attack to modify the time of the last usages of the card

not possible with our antenna. A current student in the lab is now working on the mechanisms to enhance the eavesdropping/skimming attacks' range.

Further studies can be done on OPUS cards to reveal the details of the authentication/encryption protocols and the possible weaknesses in the design. One of the main problems we faced in this study was the fact that . For every communication session that was recorded, we needed to pass a metro/bus station with our Proxmark3's antenna attached to the OPUS card. This problem could have been eased by developing a more powerful antenna for the Proxmark3 which required more time and This means that we could not record a large number of records automatically as one can do with a reader in the lab. If the reader was available, the following attempts could lead to more advanced attacks against the system:

- The best explanation for sending the date/time of the last three usages of the card in clear and before the authentication might be that this information is used as a source of entropy for the reader/tag to be used in the authentication process. Although this explanation is not a good excuse for sending private information in clear, a differential cryptanalysis assuming that nonce is a function of the last three usages may reveal more information about the system. To analyze the system from that aspect, one needs to take each chunk of data (time - date - station) as a variable and study the difference in the nonce when each of these variables is slightly changed.
- Another possibility is that the nonce is a function of the ID of the tag, printed on the back and used in the anti-collision process. To check the possibility of any dependence between the ID and the nonce, one can buy several cards

together (and because of the serial generation decision, he gets cards with very close IDs) and analyze the nonce generated with every card.

- Record thousands of traces of communication between a single reader and a card in a constant state. If a pseudo-random number generator is used in the card/reader whose seed length is less than 16 bits one should be able to see a repetition in the nonce used by the card/reader in less than 65,536 traces of communications recorded. Since every transaction takes about 200 millisecond, this experiment will take less than 4 hours to be complete.
- There is always the possibility of hardware reverse engineering the card, by removing the top layers of the chip with acetone and study the VLSI pattern of the hardware implementation of the protocol. The hardware implementation gives good hints about the protocol, which can be followed later through software analysis for more detailed information.
- Study how the system responds to frequency jamming at different steps of the communication. Try to interfere into the communication after every round of challenge/response to see if the system can handle the failure or it crashes or causes a de-synchronization between the reader and the tag.

These techniques might give some hints about the possible flaws in the authentication system. Decision for further analysis can be made based on the results of these techniques.

3.6 Impacts and Recommendations

Privacy protection in an information system is concerned about when, who and why some piece of information is collected or destroyed about a person. It is also

concerned about how the collected information is protected against accidental or deliberate disclosure to unauthorized party. This research shows that, the OPUS card design does not satisfy these requirements. An adversary can easily identify and/or track a person with the help of his OPUS card. Furthermore, the adversary can even track the bus lines/metro stations that the card holder uses for daily commute to get more information about where he lives, where he works or more generally where he has been and design a more advanced attack against him based on this information. Unfortunately, this privacy violation can be done with minimum equipment and technical knowledge, without the user realizing that the information is been accessed and with no "insider" knowledge (cryptographic keys or passwords) about the system. Referring to the pervasive use of OPUS cards in the island of Montreal, the privacy issues can affect a very large group of citizens and it merits more effort from the responsible parties to solve the problem.

If for some reason, this information "absolutely" needs to be accessible without protection, at the very least, users should be informed. The card holders should know that their OPUS card is holding certain pieces of information and is ready to send them anytime, if asked with RF signals in proximity, without their knowledge or consent.

Therefore, our recommendations regarding the privacy violation issues in OPUS cards are:

Recommendation 1: Inform the public about the decision that has been made about compromising the privacy for whatever reason. Tell the card holders why this information is being collected, what it will be used for, who will be able to see it, how

it will be protected, the consequences of not providing the information, and rights of the card holders if the policy is violated. The individuals can then decide whether to use the card in full knowledge of risks [63].

Recommendation 2: Contact the vendor to request adequate solutions to these privacy problems, or otherwise consider the adaptation of alternate technologies.

Recommendation 3: Continue the research on the security of the authentication protocol, in order to validate how secure the systems is against common frauds.

CHAPTER 4
Recommendations for Future RFID Systems

For this thesis, we studied several attacks designed against different RFID systems, including TI DST, Mifare classic and HID iClass. We also partially analyzed OPUS cards for security and privacy weaknesses. It seems that most of the current RFID systems are common in a set of specifications that make them vulnerable against different types of attacks. We think that the following recommendations can improve these systems in terms of the security and privacy metrics. The recommendations have been categorized into 5 groups of software update, hardware update, new card, new technology and new policy, in terms of the cost and the disruption of service that the application of the modifications imposes to the current systems.

F1. One of the basic problems of almost all RFID systems is that they use the static serial number of the tag (CSD) in anti-collision phase at the beginning of the communication. This CSD is unique and is sent in clear and hence can be used to identify the tag and/or the card holder. Our recommendation is to use a random/pseudorandom number that is different every time in the anti-collision process. This way, the anonymity of the tag/card holder will be preserved. This can be accomplished by:

F1-1. Study a lightweight pseudo random number generator specifically designed for low-cost RFID systems and use it to generate a dynamic ID for the tag in the anticollision phase, which can also be used to generate nonce for the

tag in the authentication phase. To implement a pseudo-random number generator we may need to update both the hardware and the software.

F1-2. Use one of the update rules used in some of the proposed authentication algorithms to refresh the dynamic ID of the tags each time after a successful authentication.

Fortunately, the second approach has shown to be very low cost (in terms of memory and computation) and feasible for passive RFID tags, therefore, it can be taken seriously for the next generation of RFID systems. Re-configuration of the already working RFID systems needs the software implementation of the update rules and an extra memory equal to the length of the ID to avoid desynchronization between the reader and the tag.

F2. If for any reason the designers are not capable of generating a dynamic ID to use in the anti-collision process, they should be careful about how the ID is generated. Since this bit stream is sent in clear, it should contain absolutely no information about the tag/card holder. Designers should minimize the information leakage from the ID to avoid the privacy problems like what happens in OPUS cards. A software update is required to apply this modification.

F3. If it is inevitable to make some pieces of information accessible without protection, it is important to make sure that the owner is aware of this information leakages, the user knows who will have access to it and what the consequences of not providing it to the system are. The party that gets access to the information should legally accept the responsibility of protecting it from unauthorized parties. This is a modification in the manufacturers policy.

F4. Companies should release their authentication and encryption algorithms and stop depending on obscurity to get security. They should let the scientists investigate their algorithms and discuss their strengths and weaknesses and probably find the vulnerabilities before the adversaries do. The previous attacks on Mifare, DST and HID iClass show that hiding the algorithm from the public is not the best way of protecting the system.

F5. When it is possible in terms of the operational cost, it is beneficial to have a button/Faraday sleeve on the cards to ask the card holder for permission before sending any information to an RF receiver. The benefits of a button or a Faraday sleeve is several folds:

F5-1. the card holder can protect its RFID card from skimming attacks.

F5-2. the card will be protected against unwanted brute force attacks

F5-3. the card holder can not deny the fact that he has used his RFID card later on; for example, in the case of electronic payments the payer can not deny that he has bought the good and excuses such as automatic payments or unwanted signal transfer will be unacceptable.

The cards should be replaced by the new cards and a minor modification in the software and the hardware is required to add a button to the operating cards. The manufacturers can also encourage the users to keep their cards in the Faraday sleeves for more security and privacy.

F6. We might investigate the possibility of using a low cost battery for RFID tags. If the limitation of power sources and computation capacities of the tags can be solved, then we can improve the authentication and encryption algorithms.

If the design process is successful, the cards should be replaced by new cards, including a battery and a new hardware to get advantage of this modification.

F7. Collect only the minimum data required to perform the transaction. This is a very reasonable principle specially when the communication channel is not secure as in the case of RFID systems. The software update and a change of policy is required to apply this modification.

F8. Keep only encrypted information on the memory of the card to make sure that if the adversary can get physical access to the card, he can not retrieve the private information from the memory. This modification requires an update in the software and a change of policy in the manufacturing company.

4.1 Near Field Communication (NFC) as an alternative

NFC is a wireless communication technology for smartphones to send or receive data with another NFC enabled device in proximity (up to 4 inches). Nokia, Philips and Sony founded the NFC forum in 2004, where communication protocols and data exchange formats for NFC enabled devices are being established. These standards are based on the existing RFID ones including ISO/IEC 14443 and Felica.

The majority of blackberry devices have NFC chips and can already be used in Canada for electronic payments. NFC enabled smartphones can be used to integrate all RFID cards in user's wallet (for different applications such as payment, metro pass, access control etc.). Unfortunately, because of the compatability issues, the NFC enabled smartphones currently use the exact same protocol that's being used on the RFID systems. However, with the growth of NFC users the companies will have the incentive to configure their system to get advantage of the more power and

computation sources available on smartphones for heavier security related computations.

However, even at the current stage, the two main feature of:

- communication only with the devices at very short distances (up to 4 inches)
- and turn on/off button for the NFC communications

can provide more privacy in comparison to current RFID systems.

Further studies can be done to design specific communication protocols for NFC devices, to get advantage of the communication/computational capacities available on a smartphone to provide a stronger security and privacy level for the system (Table 4-1).

	software update	hardware update	new card	new tech.	new policy
light pseudo-random number generator	✓	✓			
update ID in anticollision	✓				
zero-knowledge ID in anticollision	✓				
inform public about info gathering					✓
publicly release the algorithms					✓
button	✓	✓	✓		
Faraday sleeve					✓
low cost battery		✓	✓		
minimum data transfer	✓				✓
store encrypted data	✓				✓
NFC as an alternative				✓	

Table 4–1: Recommendations for more secure RFID systems

CHAPTER 5
Conclusions

The aim of this thesis has been to improve the security and privacy mechanisms of RFID systems through developing the methodology for analysis of the RFID systems that had not been studied before, in the hope to find the possible flaws before the adversaries do. We also expressed our hope to find a set of design modifications that improve the security and privacy of RFID systems, in the introduction chapter. In Chapter 2, we studied the previous attacks against TI DST, Mifare Classic and HID iClass, to summarize the methodology of the attacks and the vulnerabilities that have been used to break the systems. In the same chapter, we studied several authentication algorithms including LMAP, SASI and Gossamer (in the group of basic operation-based authentication algorithms) and T2MAP and HB family protocols (in the group of symmetric-key authentication algorithms). We discussed the strengths and the weaknesses of these algorithms along with the attacks that have been designed against them, to show that the problem of designing a provably secure RFID system is still unsolved.

In Chapter 3, we discussed our approach to analyze the OPUS cards, used in the city of Montreal for access control to the public transportation service. We showed that in OPUS cards

1. the static UID, that is printed on the back of the cards, is generated sequentially and is sent in clear as part of the anticollision process

2. the date, the time and the station of the last three usages of the card is sent in clear before the authentication process

We also discussed that an adversary can use these privacy violating specifications to track the card holder in an automatic way and without his consent. He can also skim the victim's card to get information about where he lives or where he works or more generally about where he has been. Moreover, the design flaws in the system such as the transaction commitment mechanism makes a set of ticket fraud attacks possible against the system.

Chapter 4 includes a set of design modifications that are proposed to improve the security and privacy of current RFID systems. These modifications are classified into 5 groups of software updates, hardware updates, new card, new technology and new policy. This classification is based on the cost and the disruption that the application of the modifications will impose to the current RFID systems manufacturing process. These recommendations include

1. design a pseudo-random number generator compatible with RFID systems restrictions,
2. use a variable UID in the anticollision process,
3. use a randomly (or pseudo-randomly) generated UID for the anticollision process,
4. publicly release the authentication and encryption algorithms that are used in the system,
5. store only encrypted information on the memory of the card,
6. study a low cost battery specifically designed for the RFID systems,

7. study the possibility of using a button or Faraday sleeve on the cards,
8. follow the minimum unprotected data transfer principle in the protocol design,
9. inform the public about any privacy violation in the system, and
10. consider Near Field Communication (NFC) as an alternative for the current RFID systems.

These modifications are not proved to be necessary nor sufficient to build a secure RFID system. However, they show a promising approach toward this goal.

APPENDIX A
PenguinInterface.java

```
import java.io.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.DefaultTableCellRenderer;
import javax.swing.table.TableColumn;
import java.awt.*;
public class PenguinInterface{
        public static void main(String[] args){
                javax.swing.SwingUtilities.invokeLater(new Runnable() {
                        public void run() {
                                createAndShowGUI();
                        }
                });
        }
public static String removeSpaces(String s) {
  StringTokenizer st = new StringTokenizer(s," ",false);
  String t="";
  while (st.hasMoreElements()) t += st.nextElement();
        return t;
}
/*
*extracts the unique ID from the anticollision messages
*The input is a string of hexadecimal message (without space)
*The output is a string representing the decimal value of the UID
*The decimal value is printed on the back of the card as well.
*/
public static String PrintID(String ID){
        String realID = removeSpaces(ID);
        String rRealID1 = "";
```

```
        String rRealID2 = "";
        // treats the UID as 2 parts of 2 bytes each in hexadecimal format
        for(int i=2; i<6; i++){
                rRealID1+=realID.charAt(i);
        }
        for(int i=6; i<10; i++){
                rRealID2+=realID.charAt(i);
        }
        //get the integer value of the hexadecimal number
        int i1 = Integer.parseInt(rRealID1,16);
        int i2 = Integer.parseInt(rRealID2,16);
        String res = (i1*Math.pow(16,4))+i2+"";
        StringTokenizer st = new StringTokenizer(res,".",false);
        String t="";
        //converts the result back to string
        while (st.hasMoreElements()) t += st.nextElement();
        StringTokenizer st2 = new StringTokenizer(t,"E9",false);
        String t2="";
        while (st2.hasMoreElements()) t2 += st2.nextElement();
        return(t2+" ");
}
/*
*extracts the data and the line of last usage from the messages
*the input is a string of hexademical message without space
*the output is a human reable form of the date and the time of the usage
*/
public static String PrintDateAndLine(String str){
        String realID = removeSpaces(str);
        String timeString = "";
        //extracts the part representing the date and the time
        for(int i=4; i<10; i++){
                timeString+=realID.charAt(i);
        }
        //1/1/1997  12:00:00 AM  "time" is the number of minutes from this time in 1997
        int timeDate = Integer.parseInt(timeString,16);
```

```
        int year = (timeDate/1024)/365;
        int month = (((timeDate/1024)-(year*365))/30)+1;
        int day = (timeDate/1024)-(year*365)-((month-1)*30)-1;
        int timeTime = (timeDate\%1024)*2;
        int hour = timeTime/60;
        int minute = timeTime\%60;
        return(year+1997+"/"+month+"/"+day+"  "+hour+":"+minute+":00"+"  "));
}
/*
*draws a table with three columns: Time, Line and Station
*/
private static void createAndShowGUI() {
        String[] columnNames = {"Time","Line", "Station"};
        String[][] data = new String[4][3];
        JPanel panel;
        JTable table;
        JFrame frame = new JFrame("Penguin");
        panel=new JPanel();
        panel.setSize(400,400);
        table=new JTable(data,columnNames);
        table.setAutoResizeMode(\(JTable.AUTO_RESIZE_OFF\));
        table.setSize(400,400);
        (table.getColumnModel().getColumn(0)).setPreferredWidth(150);
        (table.getColumnModel().getColumn(1)).setPreferredWidth(150);
        (table.getColumnModel().getColumn(2)).setPreferredWidth(150);
        TableColumn col0 = table.getColumn(0);
        DefaultTableCellRenderer cellrenderer = new DefaultTableCellRenderer();
        cellrenderer.setFont(new Font("Arial", Font.BOLD, 15));
        panel.add(table);
        frame.getContentPane().add(panel);
        JTextField tf;
        frame.setDefaultCloseOperation(\(JFrame.EXIT_ON_CLOSE\));
        File file = new File("C:\\rfid\\pm3-trunk\\client\\Results.txt");
        FileInputStream fis = null;
        BufferedInputStream bis = null;
```

```
DataInputStream dis = null;
if(table.isCellEditable(0,1))
        {
                table.editCellAt(0,1);
                tf = (JTextField)table.getCellEditor().
                getTableCellEditorComponent(table, "Date and Time", false, 0, 1);
                tf.requestFocus();
                tf.setSelectionStart(0);
                tf.setSelectionEnd(tf.getText().length());
        }
if(table.isCellEditable(1,0))
        {
                table.editCellAt(1,0);
                tf = (JTextField)table.getCellEditor().
                getTableCellEditorComponent(table, "Last Usage", false, 1, 0);
                tf.requestFocus();
                tf.setSelectionStart(0);
                tf.setSelectionEnd(tf.getText().length());
        }
if(table.isCellEditable(2,0))
        {
                table.editCellAt(2,0);
                tf = (JTextField)table.getCellEditor().
                getTableCellEditorComponent(table, "Second Last Usage", false, 2, 0);
                tf.requestFocus();
                tf.setSelectionStart(0);
                tf.setSelectionEnd(tf.getText().length());
        }
if(table.isCellEditable(3,0))
        {
        table.editCellAt(3,0);
        tf = (JTextField)table.getCellEditor().
        getTableCellEditorComponent(table, "Third Last Usage", false, 3, 0);
        tf.requestFocus();
        tf.setSelectionStart(0);
```

```
        tf.setSelectionEnd(tf.getText().length());
        }
if(table.isCellEditable(0,2))
        {
                table.editCellAt(0,2);
                tf = (JTextField)table.getCellEditor().
                getTableCellEditorComponent(table, "Line and Station", false, 0, 2);
                tf.requestFocus();
                tf.setSelectionStart(0);
                tf.setSelectionEnd(tf.getText().length());
        }
try{
fis = new FileInputStream(file);
bis = new BufferedInputStream(fis);
dis = new DataInputStream(bis);
int i = 0;
int dcml;
String whatToShow = "";
while (dis.available() != 0) {
        if(i==0){
                whatToShow=PrintID(dis.readLine());
                if(table.isCellEditable(0,0))
                {
                        table.editCellAt(0,0);
                        tf = (JTextField)table.getCellEditor().
                        getTableCellEditorComponent(table, whatToShow, false, 0, 0);
                        tf.requestFocus();
                        tf.setSelectionStart(0);
                        tf.setSelectionEnd(tf.getText().length());
                }
        }
        else if(i==1){
                whatToShow=PrintDateAndLine(dis.readLine());
                if(table.isCellEditable(1,1))
                {
```

```
                table.editCellAt(1,1);
                tf = (JTextField)table.getCellEditor().
                getTableCellEditorComponent(table, whatToShow, false, 1, 1);
                tf.requestFocus();
                tf.setSelectionStart(0);
                tf.setSelectionEnd(tf.getText().length());
        }
}
else if(i==2){
        whatToShow=PrintDateAndLine(dis.readLine());
        if(table.isCellEditable(2,1))
        {
                table.editCellAt(2,1);
                tf = (JTextField)table.getCellEditor().
                getTableCellEditorComponent(table, whatToShow, false, 2, 1);
                tf.requestFocus();
                tf.setSelectionStart(0);
                tf.setSelectionEnd(tf.getText().length());
        }
}
else if(i==3){
        whatToShow=PrintDateAndLine(dis.readLine());
        if(table.isCellEditable(3,1))
        {
                table.editCellAt(3,1);
                tf = (JTextField)table.getCellEditor().
                getTableCellEditorComponent(table, whatToShow, false, 3, 1);
                tf.requestFocus();
                tf.setSelectionStart(0);
                tf.setSelectionEnd(tf.getText().length());
        }
}
else{
        System.out.println("Extra Lines Here");
}
```

```
                i++;
        }
        }
        catch(Exception e){
                e.printStackTrace();
                System.out.println("No File Found");
        }
        frame.setSize(400, 400);
        frame.getContentPane().add(panel);
        frame.pack();
        frame.setVisible(true);
        }
}
}
```

REFERENCES

[1] B. Glover and H. Bhatt, "RFID Essentials", O'Reilly Media, 2006.

[2] TrackIt Systems Archive, "Active and Passive RFID: Two Distinct, But Complementary, Technologies for Real-Time Supply Chain Visibility", 2002.

[3] K. Finkenzeller, "RFID Handbook: Fundamentals and Applications in Contactless Smart Cards, Radio Frequency Identification and Near-Field Communication", Wiley, 2010.

[4] J. Gordon, U. Kaiser and T.A. Sabetti, "A Low Cost Transponder for High Security Vehicle Immobilizers", 29^{th} International Symposium on Automative Technology and Automation (ISATA), 1996.

[5] A. Juels, "RFID Security and Privacy: A Research Survey", Selected Areas in Communications, IEEE Journal 24.2, 2006.

[6] S. Garfinkel, A. Juels and R. Pappu, "RFID Privacy: An Overview of Problems and Proposed Solutions", IEEE Security and Privacy, 2005.

[7] M. Rieback, B. Crispo and A. Tanenbaum, "The Evolution of RFID Security", IEEE Pervasive Computing 5.1, 2006.

[8] D. Molna and D. Wagner, "Privacy and Security in Library RFID: Issues, Practices, and Architectures", 11^{th} ACM Conference on Computer and Communications Security, 2004.

[9] P. Rotter, "A Framework for Assessing RFID System Security and Privacy Risks", IEEE Pervasive Computing 7.2, 2008.

[10] Allianz Canada Archive, "Allianz Canada Encourages Customers to Fight Auto Theft", 2001, http://www.insurance-canada.ca/market/canada/Allianz200107.php, A.T: July 2012.

[11] P. Peris-Lopez, J. C. Hernandez-Castro, J. M. Estevez-Tapiador and A. Ribagorda, "EMAP: An Efficient Mutual Authentication Protocol for Low-cost RFID Tags", On the Move to Meaningful Internet Systems (OTM), Springer Berlin/Heidelberg, 2006.

[12] P. Peris-Lopez, J. C. Hernandez-Castro, J. M. Estevez-Tapiador and A. Ribagorda, "M2AP: A Minimalist Mutual-Authentication Protocol for Low-cost RFID Tags", Ubiquitous Intelligence and Computing, 2006.

[13] P. Peris-Lopez, J. C. Hernandez-Castro, J. M. Estevez-Tapiador and A. Ribagorda, "LMAP: A Real Lightweight Mutual Authentication Protocol for low cost RFID tags, 2^{nd} Workshop on RFID Security, 2006.

[14] Texas Instruments Archive, "Low Frequency Micro Evaluation Kit, Reference Guide", 2001, http://www.ti.com/lit/ug/scbu040/scbu040.pdf, A.T: November 2011.

[15] H. Y. Chien, "SASI: A New Ultra Lightweight RFID Authentication Protocol Providing Strong Authentication and Strong Integrity", Dependable and Secure Computing, IEEE Transactions 4.4, 2007.

[16] M. R. Sohizadeh Abyaneh, "Passive Cryptanalysis of the UnConditionally Secure Authentication Protocol for RFID Systems", Cryptology ePrint Archive, 2010.

[17] H. M. Sun, W. C. Ting and K. H. Wang, "On the Security of Chien's Ultralightweight RFID Authentication Protocol", Dependable and Secure Computing, IEEE Transactions 8.2, 2011.

[18] P. Peris-Lopez, J. C. Hernandez-Castro, J. M. Estevez-Tapiador and A. Ribagorda, "Advances in Ultralightweight Cryptography for Low-cost RFID Tags: Gossamer Protocol", Workshop on Information Security Applications, 2008.

[19] C. Kim, E. Jung, D. H. Lee, C. Jung and D. Han, "Cryptanalysis of INCrypt32 in HID's iCLASS Systems", Cryptology ePrint Archive, Report 2011/469, 2011.

[20] A. Klimov and A. Shamir, "Cryptographic Applications of T-functions", Selected Areas in Cryptography, Springer Berlin/Heidelberg, 2004.

[21] S. Boumerdassi, P. K. Diop, E. Renault and A. Wei, "T2MAP: A Two-Message Mutual Authentication Protocol for Low-Cost RFID Sensor Networks", IEEE Vehicular Technology Conference (VTC), 2006.

[22] N. J. Hopper and M. Blum, "Secure Human Identification Protocols", Asiacrypt, 2001.

[23] Statistics Canada, "Commuting Patterns and Places of Work of Canadians", 2006 Census, http://www12.statcan.ca/census-recensement/2006/as-sa/97-561/p2-eng.cfm, A.T: May 2011.

[24] C. Dewolf, "Meet OPUS: your new smart card", http://spacingmontreal.ca/2010/04/, A.T: May 2011.

[25] A. Steil, "Introduction to OpenBTS", http://www.fh-kl.de/∼ andreas.steil, A.T: May 2011.

[26] M. Bottino, "On The Security and Privacy Analysis of RFID Systems", Master thesis, Ecole Polythecnique de Montreal, 2009.

[27] A. Juels and S. A. Weis, "Defining Strong Privacy for RFID", Pervasive Computing and Communications Workshops (PerCom), 2007.

[28] F. D. Garcia, G. de Koning Gans, R. Muijrers, P. van Rossum, R. Verdult, R. W. Schreur and B. Jacobs, "Dismantling Mifare Classic", Computer Scurity−ESORICS, 2008.

[29] T. Li, "Vulnerability Analysis of EMAP-an Efficient RFID Mutual Authentication Protocol", International Conference on Availability, Reliability and Security (ARES), 2007.

[30] M. Barasz, B. Boros, P. Ligeti, K. Loja and D. A. Nagy, "Passive Attack Against the M2AP Mutual Authentication Protocol for RFID Tags", 1^{st} International EURASIP Workshop on RFID Technology, 2007.

[31] K. Nohl and D. Evans, "Reverse-Engineering a Cryptographic RFID Tag", 17^{th} USENIX Symposium, 2008.

[32] Calypso Networks Associations Archive, "The Calypso Handbook", http://www.calypsonet-asso.org, A.T: June 2011.

[33] G. P. Hancke, "Practical Attacks on Proximity Identification Systems", IEEE Symposium on Security and Privacy, 2006.

[34] S. Bono, M. Green, A. Stubblefield, A. Rubin, A. Juels and M. Szydlo, "Analysis of the Texas Instruments DST RFID", http://securityevaluators.com/content/case-studies/tiris/index.jsp, A.T: June 2012.

[35] P. Peris-Lopez, J. C. Hernandez-Castro, J. M. Estevez-Tapiador and A. Ribagorda, "An Efficient Authentication Protocol for RFID Systems Resistant to Active Attacks", Emerging Directions in Embedded and Ubiquitous Computing, 2007.

[36] E. Zenner, "Authentication for RFID Tags: Observations on the HB Protocols", 4^{th} Interdisciplinary Seminar on Applied Mathematics, 2009.

[37] Y. Yousuf and V. Potdar, "A Survey of RFID Authentication Protocols", Advanced Information Networking and Applications (AINAW), 2008.

[38] K. Nohl, "Cryptanalysis of Crypto-1", http://www.cs.virginia.edu/∼kn5f/Mifare.Cryptanalysis.htm, A.T: July 2012.

[39] "USRP purchasing page", http://www.ettus.com/order, A.T: August 2011.

[40] "Proxmark3 Homepage", http://proxmark3.com/, A.T: August 2011.

[41] R. Anderson and M. Kuhn, "Low Cost Attacks on Tamper Resistant Devices", International Workshop on Security Protocols, Springer Berlin/Heidelberg, 1997.

[42] S. H. Weigart, "Physical Security Devices for Computer Subsystems: A Survey of Attacks and Defences", Workshop on Cryptographic Hardware and Embedded Systems (CHES), 2000.

[43] P. Peris-Lopez, "Lightweight Cryptography in Radio Frequency Identification (RFID) Systems", PhD Thesis, Universidad Carlos III de Madrid, 2008.

[44] M. Bruso, K. Chatzikokolakis and J. den Hartog, "Formal verification of privacy for RFID systems", Computer Security Foundations Symposium (CSF), 2010.

[45] T. Li and G. Wang, "Security Analysis of Two Ultra-Lightweight RFID Authentication Protocols", New Approaches for Security, Privacy and Trust in Complex Environments, 2007.

[46] J. C. Hernandez-Castro, J. M. Estevez-Tapiador, P. Peris-Lopez and J. Quisquater, "Cryptanalysis of the SASI Ultralightweight RFID Authentication Protocol with Modular Rotations", Technical Report, 2008.

[47] S. A. Weis, "Security and Privacy in Radio-Frequency Identification Devices", Masters Thesis, Massachusetts Institute of Technology, 2003.

[48] A. Juels and S. Weis, "Authenticating Pervasive Devices with Human Protocols", Advances in Cryptology (CRYPTO), Springer Berlin/Heidelberg, 2005.

[49] Z. Bilal, A. Masood and F. Kausar, "Security Analysis of Ultra-lightweight Cryptographic Protocol for Low-cost RFID Tags: Gossamer Protocol", International Conference on Network-Based Information Systems (NBIS), 2009.

[50] J. Gilmore, "Cracking DES: Secrets of Encryption Research, Wiretap Politics and Chip Design", O'Reilly, 1998.

[51] Calypso Functional Specification, "Card Application, Ref: 010608-NT-CalypsoGenSpecs", http://www.calypsostandard.net/index.php, A.T: July 2012.

[52] H. Gilbert, M. Robshaw and H. Sibert, "An Active Attack Against HB+, A Provably Secure Lightweight Protocol", Electronics Letters 41.21, 2005.

[53] ProxClone Archive, "iClass Card Cloning Using an RW300 Reader/Writer", http://proxclone.com/pdfs/iClass−Cloner−rev0.pdf, A.T: March 2012.

[54] S. C. Bono, M. Green, A. Stubblefield, A. Juels, A. D. Rubin and M. Szydlo, "Security Analysis of a Cryptographically Enabled RFID Device", 14^{th} USENIX Security Symposium, 2005.

[55] F. D. Garcia, G. de Koning Gans, R. Verdult, "Exposing iClass Key Diversification", 20^{th} USENIX Security Symposium, 2011.

[56] J. Bringer, H. Chabanne and E. Dottax, "HB++: a Lightweight Authentication Protocol Secure Against Some Attacks", Security, Privacy and Trust in Pervasive and Ubiquitous Computing (SecPerU), 2006.

[57] S. Piramuthu, "HB and Related Lightweight Authentication Protocols for Secure RFID Tag/Reader Authentication", Conference on Collaborative Electronic Commerce Technology and Research (CollECTeR), 2006.

[58] "HID Global Smart Cards Webpage", http://www.hidglobal.com/technologyGeneral.php?id=2, A.T: March 2011.

[59] HID Corporation, "HID iClass Readers and Cards, Application Note Number 28", 2009.

[60] M. Meriac, "Heart of Darkness, exploring the uncharted backwaters of HID iCLASS security", 27^{th} Chaos Communication Congress, 2010.

[61] About.com, "STM Montreal - Overview of Montreal Transit Use", http://montreal.about.com/od/gettingaroundtown/ss/stm-metro.htm, A.T: March 2012.

[62] Canadian Broadcasting Corporation (CBC), "Smart Transit Cards Come to Montreal, Quebec", http://www.cbc.ca/news/technology/story/2008/04/21/qc-smartcards0421.html, A.T: March 2012.

[63] Smart Card Alliance, "Privacy and Secure Identification Systems: The Role of Smart Cards as a Privacy-Enabling Technology", White Paper, 2003.

[64] A. Juels, D. Molnar and D. Wagner, "Security and Privacy Issues in E-passports", 1^{st} International Conference on Security and Privacy for Emerging Areas in Communications Networks, 2005.