Abstract

Oyster Card, as a form of electronic ticketing, is now widely being used on public transport services in London underground. MIFARE Classic system, based Oyster card, was used as a transport system in London before 2009. However, its security has been proved to be weak by a group of researchers from Radboud University in Holland. As an alternative, from 2009 until now, MIFARE DESFire is now being used in London.

The main aims of this project are to investigate and analyze the potential security risks against personal information within Oyster card system, and then give some security countermeasures based on the analysis.

The following summarizes the main contributions and achievements in this project.

- Give background knowledge and technologies related with the Oyster card system, such as system security requirements, attacks and threats analysis and cryptography background, see section 2.
- Describe the technical details about MIFARE Classic, especially the communication principle and authentication protocol between cards and terminals. The security of MIFARE Classic strongly relies on the authentication protocol and its algorithm. Details are showed in subsection 3.1.
- Analyze the security of MIFARE Classic. It has been proved that it is able to recover keystream and then read and modify the contents of Oyster cards because of the weaknesses of PRNG and authentication protocol. Moreover, the encryption algorithm has been recovered and the secret key has been retrieved as well, due to the weak secrecy of algorithm itself and the shortness of key size. Details are described in subsection 3.2
- Compare the secure features between MIFARE Classic and MIFARE DESFire, see section 4.
- Propose some recommendations to enhance the security of Oyster card system. From the perspective of security designers, using hash function to encrypt the authentication protocol such that the communications between cards and terminals are encrypted and hard to decrypt by attackers. Also, synchronized secrets are used to detect card cloning attack. In addition, some recommendations are given from the responsibility of end users. Details are showed in section 5.

Acknowledgements

I would like to express my thanks to my supervisor, Dr. Theodore Tryfonas, for his continuous and invaluable support, encouragement and advice throughout this project.

Thanks to Professor Nigel Smart and Dr. Bogdan Warinschi for their advice and feedbacks in the poster presentation.

Thanks also to the teaching staffs in University of Bristol, especially my course director, Steve Gregory for his help and guidance throughout my Masters Course. After this year's study, not only my academic knowledge but my confidence has been largely improved.

Finally, I wish to greatly thank my family for their financial support and spiritual encouragement, especially my brother, without his constant support and encouragement, I could not finish this project.

Table of Contents

1 INTRODUCTION

1.1 Aims and Objectives

The primary aim of this project is to investigate and analyze the potential level risks again an individual’s personal information within “smart” spaces. We will select one typical case, Oyster card used as a ticketing medium for transporting in London underground. Based on the analysis, we produce some suitable security countermeasures. The suggested measures will be addressed both issues of security design of such “smart” applications, i.e. application developer's perspective and issues of an individual's personal responsibilities, i.e. end user's perspective.

1.2 Structure

This paper consists of 5 main sections. In the first section, we introduce the background and some basic knowledge about technologies related with the transport ticketing system, security requirements and analysis of attacks and threats. Also, some cryptography background knowledge is given.

Secondly, we analyze the security risks of MIFARE Classic based Oyster card, which was used as a transport ticketing medium in London underground before 2009.

Thirdly, we talk about MIFARE DESFire based on Oyster card, which replaces MIFARE Classic, being used as a transport ticketing system in London until now. We compare MIFARE DESFire with MIFARE Classic from the structure features and security features.

Fourthly, we give some recommendations to enhance the security of Oyster card system. The recommendations are proposed from both the perspective of the security designers and the perspective of end users.

Finally, the conclusion is given and then future works are discussed in terms of other considerations in evaluating and determining the security of the whole system.

2 BACKGROUND

It is admitted that the convergence and increasing of ubiquitous computing, such as sensor-based system, wireless networking, mobile and embedded device, etc. is providing tremendous opportunities for interaction design in some “smart” spaces, which provides a more efficient and convenient life for human beings. For example, due to the advantages of utilizing smart cart technologies, especially contactless smart cards, they are now widely being used in various areas, such as bank, transportation, secure entry of buildings, etc.

At the same time, the amount of personal information that ubiquitous computing systems captured and stored is rapidly increasing. Much personal information, even sensitive data, is stored in the back end databases or devices, for example, the owner’s name, data of birth, card number and balance are stored in the smart card.

In this section, we introduce some related technologies used in transport ticketing system, and then talk about security considerations for the system. Furthermore, different types of attacks and threats against smart cards and the system are given. Finally, we provide some cryptography knowledge.

2.1 Technologies in Transport Ticketing System

In this part we introduce the entities involved in the smart card transport system and the standardization of the system schemes. Then we focus on the core technology used in the system, RFID, which used for authentication and communication between contactless smart cards and the terminals.

2.1.1 Entities Involved

There are different entities involved in the smart card transport based scheme. The following are the basic ones.

- **Smart card**

A smart card is essentially a credit card sized piece of plastic card with a microchip embedded in it. The microchip can perform functions required by the card, such as storing data, processing data, and writing data. There are various kinds of cards on the market. They differ in size, casting, memory, and computing power. They also differ in the security features they provide.

Interface
When using a smart card for a particular application, one of the most important criteria that must be considered is how the smart card communicates with other devices. These are

various types of smart card, as shown in Figure 2.1.

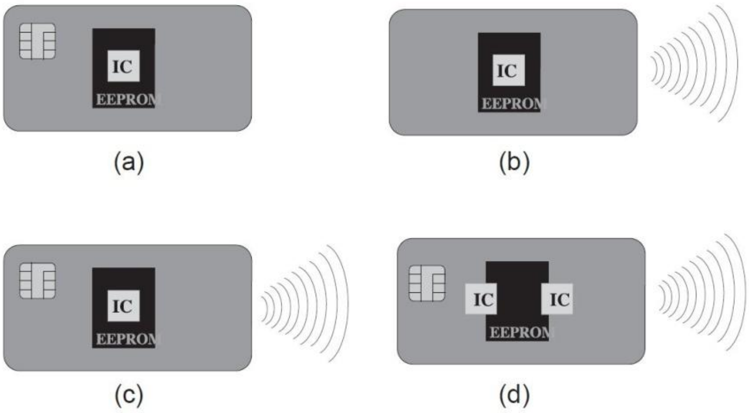


Figure 2.1. The various types of smart card. Source: Ref. [3].

(a) Standard contact card with one chip-set and one interface

(b) Standard contactless card with one chip-set and one interface

(c) Combi-card with one chip-set and two interfaces

(d) Hybrid-card with two chip-set and two interfaces

As showed in Figure 2.1, there are two basic choices of smart card interface, contact and contactless.

Contact smart card has a 3 by 5 mm security chip embedded on the surface of the card. When the card is inserted in a reading device, connection is made to each of the metal pads. Contact smart cards are now widely being used in many applications, such as in the bank to be as the digital cash (see Figure 2.2).

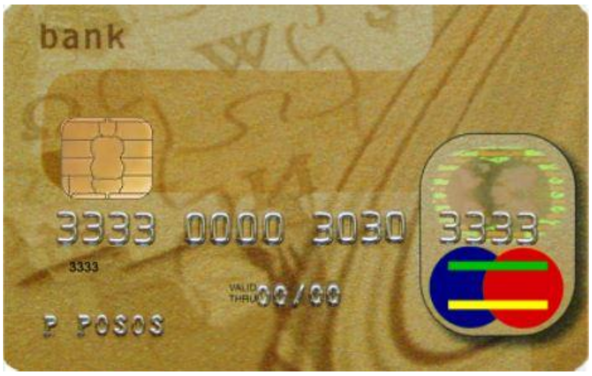


Figure 2.2. An example of contact smart card used in the bank.

A second one is contactless smart card, in which the card communicates with and is powered by the reading device through RFID technology. Most contactless smart card can be read from a distance of about 10cm, and in some cases they can be read without being removed from a wallet or purse. Apart from the difference in the communication interface, there are no other real logical differences between contactless smart cards and contact smart cards, in terms of processing power, memory and cryptographic capabilities. Due to the main characteristics of contactless smart card technology, it is being used in many kinds of applications and systems nowadays, such as public transport ticketing system (e.g. Oyster card in London underground transportation, as shown in Figure 2.3), a secure entry system for office or university buildings, etc.

Figure 2.3. Oyster card, a kind of contactless smart card, used in London underground.

Memory

There are two types of non volatile memory in smart cards. One is ROM (Read Only Memory), which holds persistent information (e.g. the smart card operating system, application) written during the manufacturing phase. The most common type of memory is normally EEPROM (Electrically Erasable Programmable Read Only Memory), which contains functions of what kinds of and how many applications the card is expected to support and all the operations such as reading data, processing data, and writing data. However, the size of the dynamic memory on a smart card is limited at present, considering the cost of EEPROM and the physical size of the memory chip within the card’s processor.

- **Card holder**

The card holder is a person to whom the card is issued. It should be noted that the card holders could be a dual role. In most cases, they do care about the security of the cards and the system because their personal information, even sensitive information, is stored in the cards and held by the system. On the other hand, under certain circumstances, they could be the originators of attacks that would result in direct or indirect benefits from finance fraud [19].

- **Card issuer**

The card issuer is the party that issues the smart cards. They have rights and abilities to control all the cards they issued.

- **Smart card manufacturer**

The smart card manufacturers are often the card distributors as they manufacture and directly deliver the cards to the transport operators.

- **Smart card application developer**

The smart card application developers are responsible for developing the smart card hardware and software including the underlying operating system.

- **Terminal**

The terminal represents the reading device that allows the card to communicate with the outside world.

- **Back office system**

The back office system, i.e. the back end database, is responsible for manipulating the card, keeping tract of the information stored in the card and back-end database.

- **Attacker**

The attacker involves any parties with an interest to attack the smart card or the security of the overall system.

2.1.2 Standardization Schemes

It is generally recognized that standardization improves interoperability of technologies, especially in international level, and enables technology suppliers to offer standardized systems. A number of standards and specifications are relevant for smart card implementations. Table 2.1 summarizes some of the major standardization activities from international level and national level.

ISO/IEC is one of the worldwide standard-setting bodies for technology, including plastic cards. The primary standards for smart cards are ISO/IEC 7816, ISO/IEC 14443, ISO/IEC 15693 and ISO/IEC 7501.

Table 2.1. Different standardization of card schemes.

Operational Level	Standard
International	ISO/IEC 7816 ISO/IEC 14443 ISO/IEC 15693 ISO/IEC 7501
National	Octopus in Hong Kong Oyster in London Charliecard in Boston

ISO/IEC 14443 is an international standard that defines the interfaces to a "close proximity" contactless smart card, including the radio frequency (RF) interface, identification, the transmission protocols for communicating and anti-collision protocols. ISO/IEC 14443 describes two types of cards, type A and type B. The main differences between the two types concern modulation methods, coding schemes and protocol initialization procedures [12].

ISO/IEC 14443 is the primary contactless smart card standard being used in various aspects, such as transportation, finance, and other access control applications. It is also being used in MIFARE technology by NXP Semiconductors, which is used in Oyster card system in London underground.

2.1.3 RFID

Radio Frequency Identification (RFID) is an automatic identification system, relying on storing and remotely retrieving data from objects using devices called "RFID tag". A RFID tag is a microchip that is capable of transmitting a unique serial number and other additional data or functions through Radio frequency (RF) signals. The goal of a RFID system is to identify objects remotely by embedding tags into the objects [5].

With the decreasing cost of equipments and tags, and increasing performance and efficiency, technologies and systems based on RFID are widely used in many aspects of lives. For example, it can be used in supply chain management or inventory control where every item can be identified and when it enters or leaves the warehouse. Also, goods in shops can be tagged in order to provide automatic theft-detection. Furthermore, RFID system can also be used to track the exactly location of people, or even more future records associated with their location, such as their activities. Oyster card, used as a transport ticketing medium in London underground, also contains a RFID tag to validate itself or deduct funds by touching it onto an electronic reader.

- **RFID components**

A RFID system is composed of three components: RFID tag, RFID reader and back-end database [5]. Figure 2.4 shows the compositions of a RFID system.

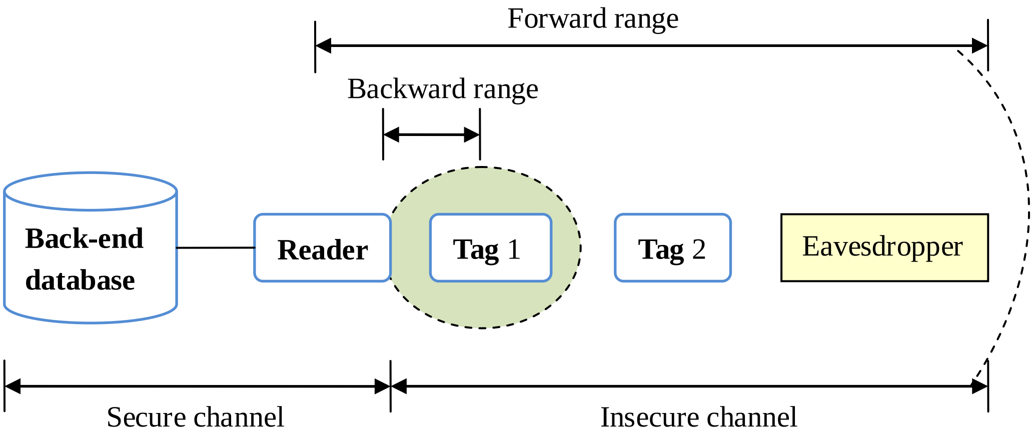


Figure 2.4. RFID system and its components [28].

The characteristics of each component are as follows.

RFID tag carries an object identifying data. When a tag receives a query from a reader, the tag transmits data to the reader using RF signals. A tag is typically composed of a microchip for storage and computation, and a coupling element, such as an antenna coil for wireless communication. Tags are divided into active tags and passive tags by the methods to be powered [26].

An active tag is often powered by a high cost on-board battery. The life span of the battery depends on the life span of the tag. The active tag can transmit data remotely. While a passive tag is inductively powered by an RF signal from the reader. It is usually used to transmit data to a short distance because tag's power is relatively lower than reader's. Since the passive tag is low cost and its life span is permanent, it is generally used in many systems and applications, including transport ticketing system.

RFID reader not only queries data from the tag, but also updates the contents of the tag through an RF interface. To provide additional functions, the reader may contain internal storage, processing power, connections to back-end database and computation, such as cryptographic calculations.

Back-end database stores and manages data received from the reader. Since it is powerful in computational capacity, it is used to computes complex computation instead of the tags or the readers.

In Figure 2.4, forward range is the area that a reader can transmit a RF signal to a tag, while backward range is the area that a tag can transmit data to a reader when it queries. Generally we assume that the communication channel between a reader and back-end database is secure. On the contrary, in the wireless communication between a reader and a tag, an adversary can monitor all the messages transmitted, so we assume this channel is insecure.

- **Challenge-response authentication**

Since the communication between tags and readers is performed at insecure channel through RF signals and also privacy problems are public concerned, so it is necessary to take some measures to secure communication and protect privacy. Since now, many kinds of security technologies for the RFID system are proposed.

One general approach is to use encryption algorithm that we will discuss in details in Section 2.4.3, by which the transmitted messages are encrypted. Another approach is to design an authentication protocol using hash function that we will talk about in Section 2.4.1. This approach can prevent an exposure of tag's ID using one-way property of hash function. The most popular proposed protocol is challenge-response authentication method, which could be based on the approaches above, encryption algorithm and hash function.

Challenge-response authentication protocols can be described that one entity (we call the prover) proves its identity to another entity (we call the verifier) by demonstrating the knowledge of a secret related with that entity, without revealing the secret itself during the protocol. This is done by providing a response to a challenge. The challenge is typically a random number chosen by one entity, and the response depends on the secret and the challenge. If the communication between the prover and the verifier is monitored, the response should not provide any useful information for subsequent identifications, as subsequent challenges are random numbers as well and will be different from the previous one.

In challenge-response by symmetric key techniques, both the prover and the verifier share a symmetric key. Figure 2.5 is an example. In larger systems with identification protocols by symmetric key techniques, to improve the security, a third trusted party is often involved to provide a common session key to two entities when every time one requests authentication with the other.

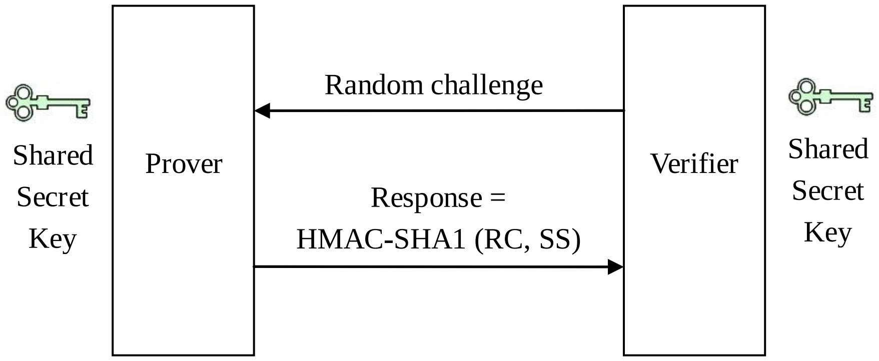


Figure 2.5. An example of challenge-response authentication based on symmetric key technique.

Challenge-response authentication based on hash function, not the encryption function, can be performed by utilizing a hash function with a shared key and challenge. When the verifier responses using a hash function to the prover, even if an adversary eavesdrops one communication, the information cannot be revealed easily because of the one-way property of hash function. Figure 2.6 shows a typical three-pass challenge-response mechanism, based on a one-way function with a shared key, provides mutual identification.

$A \rightarrow B$: r_A

$B \rightarrow A$: $r_B, h_k(r_B, r_A, A)$

$A \rightarrow B$: $h_k(r_A, r_B, B)$

Figure 2.6. An example of challenge-response authentication based on hash function.

- **Security and privacy risks**

In RFID system, since an adversary can eavesdrop one communication between a tag and a reader, personal information and privacy could be disclosed to an unauthorized reader using various methods. Therefore, RFID system must be designed to be secure against kinds of attacks such as eavesdropping, reply attack, spoofing attack, and the privacy risks should be considered as well.

Privacy risks

Information leakage: Nowadays, people are prone to carrying various tagged objects around with them, e.g. smart card as a transport ticketing medium, which contains much personal information that should be kept secure. In RFID system, the tag emits distinguishable information in response to a query form a nearby reader, and it is possible that the data in the tag could be leaked without the acknowledgement of the owner.

Traceability: when a target tag transmits a response to a reader, an adversary can record the transmitted communication and establish another link between the response and the target tag. Once the link is established, the user can be traced by tracking the ID of the tag.

Security risks

Eavesdropping: Similar to the information leakage in privacy risks, a passive adversary can easily eavesdrop one communications between a tag and a reader without user's recognition and then obtain secret information.

Replay attack: An active attacker can intercept into the process of the RFID systems actively. The adversary eavesdrops the response message from the tag by disguising as the right reader, and then disguise as the right tag to re-transmit the message to the legitimate reader.

Tampering: An adversary could tamper with a tag because low-cost tags have no tamper-resistance mechanisms. It can induce information leakage problems of a tag owner. The adversary tries to get knowledge of previous events in which that tag participates.

- **Security requirements**

To secure RFID against the threats and attacks, technical problems should be overcome. Before that, we need to provide security requirements to protect RFID privacy. The followings are some popular requirements to protect RFID privacy [24].

Data confidentiality: The private information of the tag should be kept secure to guarantee users’ privacy. The information of the tag must be meaningless if it is eavesdropped by an unauthorized reader.

Indistinguishability: Responses emitted by the tag must be indistinguishable from truly random values. Moreover, they should be irrelevant to tag’s ID. If the adversary can distinguish that particular output from a target tag, the tag can be traced.

Forward security: Even if the adversary acquires the secret data stored in the tag, it is still very hard or even not possible to trace the data back through previous events in which the tag was involved. In other words, the adversary cannot associate the current output with previous output if he or she only eavesdrops the tag’s output.

Anti-cloning: Anti-cloning is an additional security requirement, which means that the adversary cannot clone a new tag without tampering. When an adversary tampers with a tag, the information on the tag is revealed, and then it is possible to clone a new tag. However, there are still many ways to clone without tampering with tags, such as the replay attack.

2.2 System Security

2.2.1 System Level Considerations

Security is not an independent element in a whole system that should be added, instead, it is the core element in the design of the entire end-to-end solution [27]. This is an important consideration, since a properly designed system can greatly reduce the risk of compromise due to the failure of any single component.

Also, a similar idea *Privacy by Design* [4], proposed by Cavoukian, "is an approach to protecting privacy by embedding it into the design specifications of technologies, business practices, and physical infrastructures. That means building in privacy up front – right into the design specifications and architecture of new systems and processes." Based on this approach, a privacy risk management framework has been built on by Information and Privacy Commissioner of Ontario, as shown in Figure 2.7.

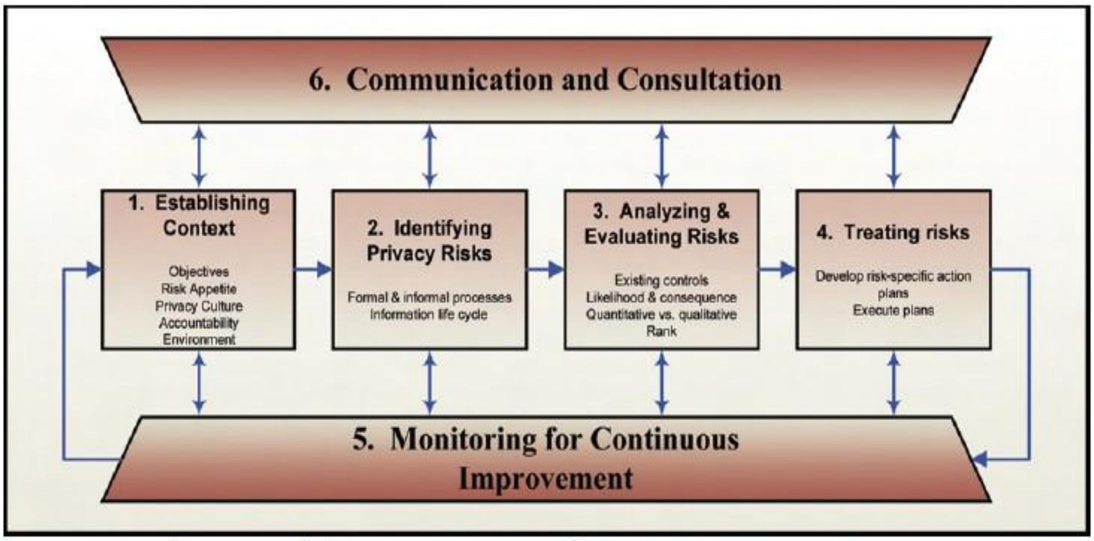

6. Communication and Consultation
1. Establishing Context
Objectives
Risk Appetite
Privacy Culture
Accountability
Environment
2. Identifying Privacy Risks
Formal & informal processes
Information life cycle
3. Analyzing & Evaluating Risks
Existing controls
Likelihood & consequence
Quantitative vs. qualitative
Rank
4. Treating risks
Develop risk-specific action plans
Execute plans
5. Monitoring for Continuous Improvement

Figure 2.7. A sample of privacy risk management framework. Source: Ref. [11].

According to Smart Card Alliance [27], security design is based on the principles of prevention, detection and reaction. Security systems should be designed with the basic assumption that someone would figure out how to attack any defenses in the system. For this reason, the design must include prevention, detection and reaction measures, as shown in Figure 2.8.

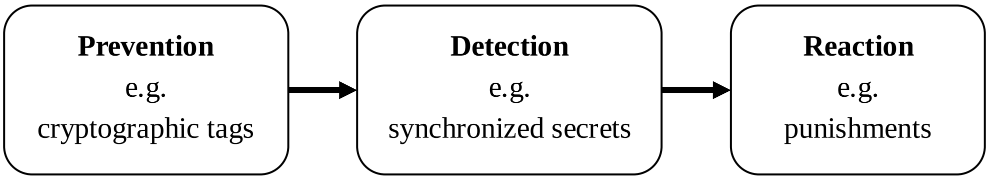


Figure 2.8. Secure considerations in designing system.

Prevention refers to the measures available to make difficult for an attacker to reveal information from the system. In the case of transportation systems, prevented measures should be taken to prevent attacking and counterfeiting the contactless smart cards that are used to pay fares, to prevent using a cloned card as a legitimate card to trick the system, or to prevent adding value to a legitimate card without paying. Preventive measures should be taken from aspects of system security, reader security and card security.

Detection of a counterfeit card that is successful in use is a critical element of security system. Detection can reduce losses into small amounts. Transportation systems, including authentication process and transaction process, should have security features and abilities to detect improper use.

Reaction refers to the response of the system owner after a successful attack is detected, which contribute an important component to the overall system security. Available measures should be taken to prevent successful attacker from repeating the process. Reaction of the system owner directly or indirectly depends on the motivated actions that minimize the negative effects for the system side and maximize the negative effects for the adversary in terms of punishment. Transportation system is supposed to have the capability of reaction to confine potential losses.

Actually, the attacks that break through a system's defenses motivate the system owner to detect the attack, thus the system developers can understand how the breach was achieved and design and deploy effective countermeasures to prevent further loss from that approach.

System-level security features should be discussed between developers and customers to understand the security analysis and potential security risks in the system. Although the perfect security cannot be achieved, a good system does reveal how they provide security and challenge legitimate hackers to try and overcome their defenses.

2.2.2 Technology Updates

Technology is constantly changing, no matter the secure technology or the attack technology, so considering security, the system needs to change as well. Many systems are hacked into on a frequent basis, if technologies or systems remain unchanged at all, they will eventually be hacked. However, when updating technologies or changing systems, precautions need to be taken, since even a slight update might open the door to a whole host of new issues. And before any changes in the system, they should be backup firstly and tested on a separate system, which allows the system to be safely checked for security flaws, especially when the system contains personal or sensitive information.

2.2.3 Secure Personal Information

In transportation ticketing system, lots of personal information, even personal privacy, has been stored in the back system office, such as personal date of birth, address, bank account, etc. in order to provide improved services. For example, with the personal information, transport agencies can implement automatic fare collection (AFC) systems that use contactless smart card technology as a payment medium. These systems are popular since they are fast, easy to access and reduce operating costs and improve efficiencies to transit operators. Oyster card system in London underground provides this service that the card will be topped up automatically when the balance of the card is blow £8. In this case, the transport agency does hold custom data e.g. bank account number, in the back system office, associated with the Oyster card number.

However, personal information could also be used in some unexpected or even malicious situations. There are numerous interviews, studies, researches and essays indicating that a general unease over the potential for abusing and lacking of control. So it is critical that transport agencies take some measures to ensure that personal information cannot be compromised, and if there is unauthorized access to systems, some measure will be taken to prevent access to all information that could be obtained.

Most application developers collect, use and store personal information, and even do some extensive data mining for potential interests. These application developers have responsibility to track and document the flow of personal information to make it clear, such as what kinds of personal information was collected and created, who interacted with it, when, where, why and by what means. Figure 2.9 shows a model of information life cycle, which ensures all the states of personal information are considered and followed from cradle to grave.

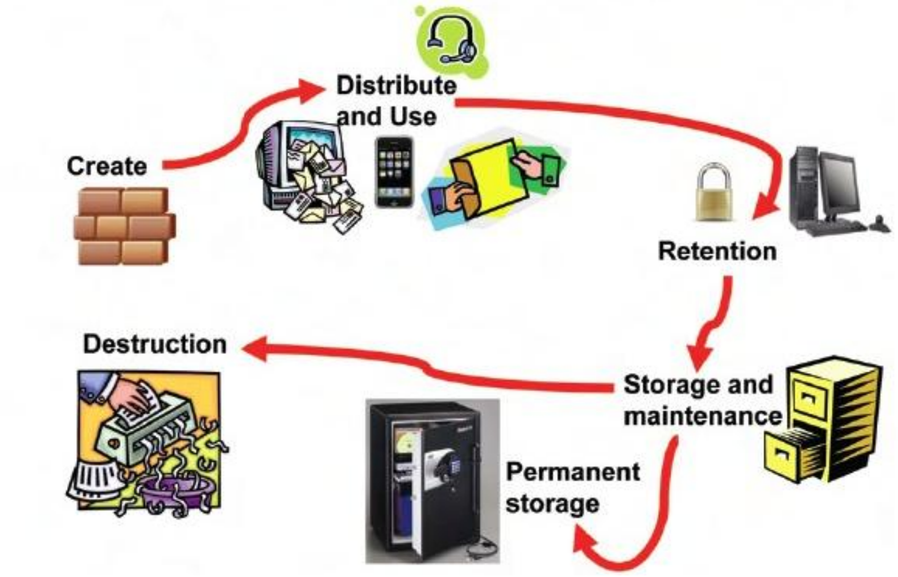


Figure 2.9. The life cycle of information. Source: Ref. [11].

Personal information is typically held at the back office system and is not stored on the card itself, which partly improve the security of data. Even if a card is broken or stolen, the personal information is still safe and secure. However, it is still possible that some attack based on the back office system data happen and personal private data are at risk.

2.3 Attacks and Threats Analysis

Smart cards have been widely used in various areas, e.g. finance, government and transport, and lots of personal information, even sensitive data, is stored in smart carts, especially when they are used as electronic mediums, such as bank cards or transport ticketing cards. Thus, smart cards are popular targets for attackers nowadays. Successful attacks enable financial fraud and obtain benefits. Besides, smart cards are cheap and easy to obtain so the attackers can easily acquire samples and practice attacking. Also smart cards are portable and the attackers can easily bring them to a hostile environment and control the conditions.

In this section, we provide some criteria to categorize the different types of attacks and threats against smart cards. A smart card attack is defined as an attempt by an entity to break through the physical or logical security of the card and then its applications or systems it depends. The aim of the attack against smart card is to obtain access to information, especially sensitive information such as account balance, which is stored in the memory of smart cards and use it to compromise the security of the card or some other entities in the smart card scheme [19].

2.3.1 Attacks Against Smart Card Components

One kind of attacks against smart cards is to focus on the microprocessor in the smart card to target the operation of cryptographic primitives, such as the generation of cryptographic keys, encryption algorithm or mutual authentication.

The attacks against smart cards can be classified into three basic categories, logical attacks, physical attacks and side-channel attacks [19].

Logical attacks

Logical attacks attempt to exploit any vulnerabilities or weaknesses in the software implementation, e.g. from the design of the smart card operating system or application. Logical attacks abuse these flaws to have access to or modify the confidential data and get benefits.

Potential logical attacks may take the form of presenting the card with many different aspects. For example, using invalid or hidden commands to retrieve data from or modify data in the smart card. A disallowed parameter value or length may also be misinterpreted and lead to unexpected results, for example, a file read command that the requested length exceed the actual file size. Moreover, the communication protocol between smart cards and readers may be insecure and it is possible for attackers to trick the smart cards into revealing secrets (Figure 2.10). Also, if cryptographic protocols that used for encrypting transaction are not well designed and less of secure, it is quite possible to perform attacks, such as replay attacks.

The advantage of this attack is that it is relatively cheap and simple to perform. Since the sensitivity of the logical attacks strongly depends on the complexity of the software, countermeasures, e.g. structured design of software and formal verification, can be take to eliminate the logical vulnerabilities.

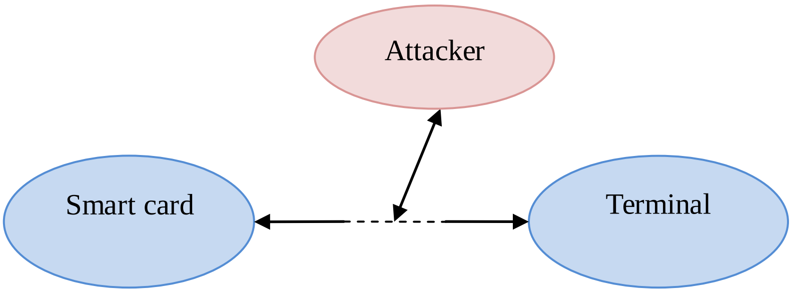


Figure 2.10. A simple example of logical attack by attacking communication protocol between smart cards and terminals.

Physical attacks

These types of attacks are performed by attempting to analyze and exploit any vulnerabilities or weaknesses in the hardware of smart cards. There are many different methods and tools to perform physical attacks. For example, Scanning Electron Microscopes (SEM) or laser cutter microscope can be used for optical analysis and reversing engineering, and then modifying the architecture of the chip, not mentioned reading the contents of the EEPROM of the card. Some of methods require low cost equipments and other may require sophisticated and expensive high-end lab equipments [19]. Compared with logical attacks, most chips may be destroyed before the attack is considered successful.

In order to prevent the physical attacks, the smart card manufactures have provided many significant improvements on the chip designs over decades, such as feature size, multi and protective layer, sensors, etc. However, due to the low cost and limited functionality of the chips, not all the advanced designs are feasible, and manufactures need to balance the costs and benefits.

Side channel attacks

Side channel attack is based on information obtained from the physical implementation of smart card system, rather than brute force or theoretical weaknesses in the algorithms. For example, timing information, power consumption, electromagnetic leaks or even sound can provide extra sources of information which can be exploited to break the system.

Some side-channel attacks require technical knowledge of the internal operation of the system, while others such as Simple Power Analysis (SPA) or Differential Power Analysis (DPA) are relatively cheap and effective to perform, and offered a high chance of success. Figure 2.11 is an example that showed an SPA trace from a typical smart card as it performs a DES operation, in which the 16 DES rounds are clearly visible [15].

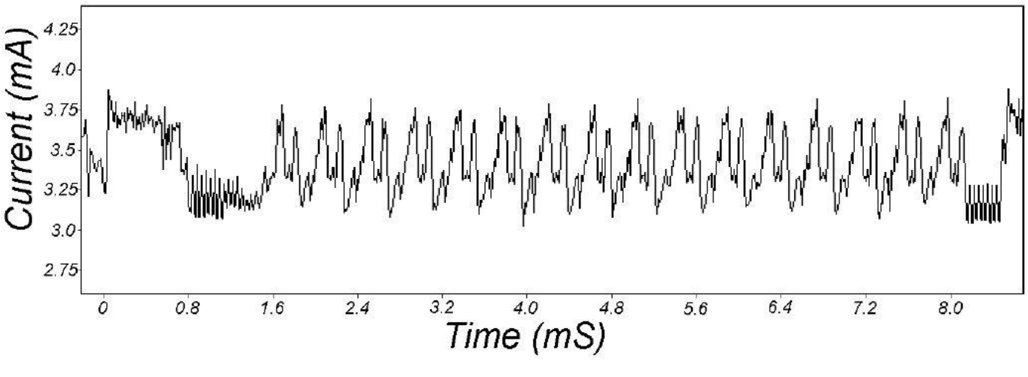


Figure 2.11. SPA trace revealing an entire DES operation. Source: Ref. [15].

Since side channel attacks rely on the emitted information, countermeasure can be taken from the aspects of reducing the release or exchange of such information, or jam the emitted channel with noise. For example, power line conditioning and filtering can help deter power monitoring attacks, and also a random delay can be added to deter timing attacks.

2.3.2 Attacks Against Terminal

Another category of attack is from the terminal side. The terminal could be stolen or cloned in order to communicate with right smart cards to receive payments. So the back-end system and the terminals need a reliable and trusted way, e.g. through a trusted third party or standard public key infrastructure (PKI), to prove authenticity to each other. Figure 11 is a typical message flow in the PKI.

2.4 Cryptography Background

2.4.1 Hash Function

A cryptographic hash function is a deterministic procedure which takes arbitrary length bit strings as input and returns fixed length bit strings as output that often called a hash value.

$$H : \{0,1\}^* \rightarrow \{0,1\}^L$$

A desirable cryptographic hash function exhibits the avalanche effect, which indicates that the output changes significantly when an input is changed slightly. Figure 2.12 is an example of one kind of cryptographic hash function, SHA-1, and it exhibits the avalanche effect.

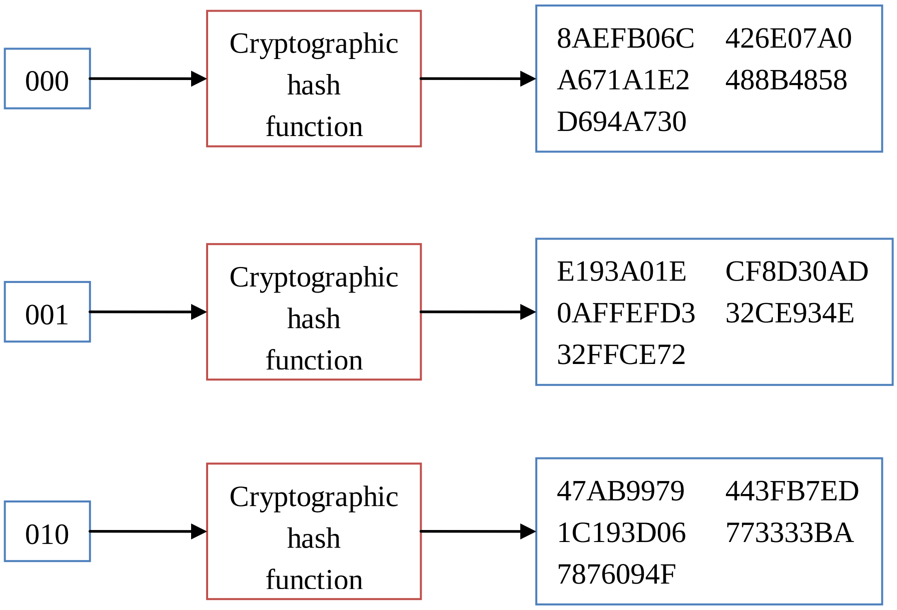

000
Cryptographic hash function
8AEFB06C 426E07A0 A671A1E2 488B4858 D694A730
001
Cryptographic hash function
E193A01E CF8D30AD 0AFFEFD3 32CE934E 32FFCE72
010
Cryptographic hash function
47AB9979 443FB7ED 1C193D06 773333BA 7876094F

Figure 2.12. An example of cryptographic hash function, SHA-1, and avalanche effect.

Usually, a cryptographic hash function should at least have the following properties in order to resistant against the popular attacks, such as exhaustive attack, birthday attack and meet in the middle attack.

- Preimage Resistance

  Given a hash value h, it should be hard to find a message m, such that $h = hash(m)$. A hash function with preimage resistance also has one-way property.

- Second Preimage Resistant

  Given one message m_1, it should be hard to find another message m_2, where $m_1 \neq m_2$, such that $hash(m_1) = hash(m_2)$.

- Collision Resistant

  It should be hard to find two different messages m_1 and m_2, such that $hash(m_1) = hash(m_2)$.

The purpose of hash function in cryptography is to provide data integrity and message authentication. Cryptographic hash functions have various applications in information security areas, notably in digital signatures, message authentication codes (MACs), and other forms of authentication. Compared with standard hash functions, cryptographic hash functions are more computational secure so that they tend to be used in contexts where it is necessary to protect data against the possibility of forgery. Also, hash function can be used in the generation of pseudorandom number and key derivation.

2.4.2 PRNG

A random number generator (RNG) is a function that generates a sequence of random numbers, and the next sequence of numbers cannot be predicted based on the previous one. Random number generation is used in wide variety of cryptographic operations, such as key generation, probabilistic encryption, challenge-response protocols and digital signature algorithms. However, in practice true randomness is hard to achieve, if it is not impossible since the computers are fully deterministic. A possible solution to this problem is proposed by the theory of cryptographically secure pseudorandom number generators (PRNG) by Yao [29].

A pseudorandom number generator is a deterministic algorithm for generating a sequence of numbers that approximates the property of randomness. The input of the generator is called the seed and the output is called the pseudorandom sequence. With each different seed, the PRNG generates a different pseudorandom sequence. With a relatively small random seed a PRNG can produce a long apparently random string. PRNG is often based on cryptographic functions like stream ciphers or block ciphers.

A requirement for PRNG is that an adversary does not know the seed has only negligible advantage in distinguishing the generator’s output sequence from a random sequence. In other words, the security of a PRNG depends on how hard it is to tell the difference between the pseudorandom sequences and truly random sequences, shortly, indistinguishability. A PRNG is said to be provably secure if distinguishing these two classes of sequences is as difficult as solving well known and supposedly mathematical hard problems [29, 1, 2]. For example, the Blum Blum Shub [1] PRNG is secure under the assumption that integer factorization is a difficult problem.

2.4.3 Encryption Algorithm

Encryption is the process of converting a plaintext message into ciphertext, which is meaningless and can be decoded back into the original message. Encryption algorithms along with secret keys are used in the encryption and decryption of data. Encryption schemes normally are based on block or stream ciphers. There are several types of encryption algorithms, the most common two are symmetric encryption and asymmetric encryption, also known as public encryption algorithm. The type and length of the keys used depend on the encryption algorithm and the security.

In symmetric encryption, both parties share the same key to encrypt and decrypt the message, as shown in Figure 2.13. To provide security and privacy, the key needs to be kept secret. If the third party knows the key, the security becomes problematic. Symmetric algorithms have the advantage of not consuming too much computing power. A few well-known examples are DES, 3DES, AES. In transport ticketing system, symmetric encryption is used because the encryption and decryption of communication between tags and readers are high-throughput application. And a large amount of cards will need to encrypted and decrypted by with only one server.

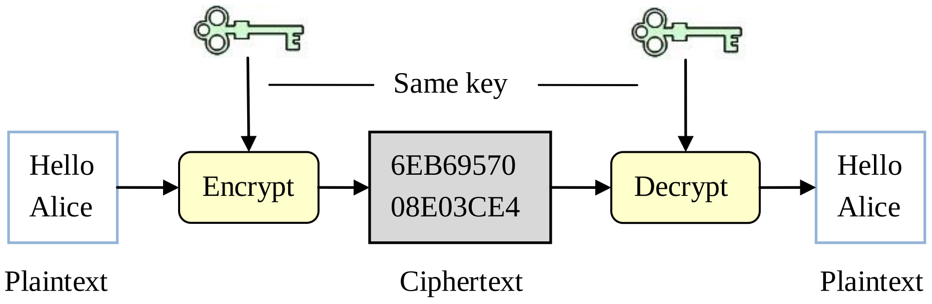


Figure 2.13. Illustration of symmetric encryption

The other kind of encryption algorithm is asymmetric encryption, in which the encryption key and the decryption key are different. One is a public key by which the sender can encrypt the message and the other is a private key by which a recipient can decrypt the message. Everybody having the public key is able to send encrypted messages to the owner of the secret key. The secret key can't be reconstructed from the public key. Well- known asymmetric algorithm are RSA, ELGAMAL. Figure 2.14 illustrates the asymmetric encryption.

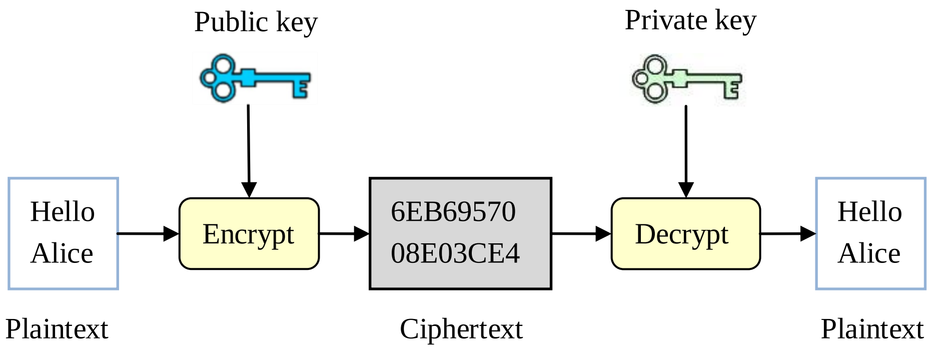


Figure 2.14. Illustration of asymmetric encryption.

Since the private key does not have to be shared, asymmetric algorithm seems more secure and to be ideally suited for real-world use. However, asymmetric algorithms are considerably slower than symmetric ones. Also since asymmetric algorithm use pairs of keys, it is more computationally expensive, which is one of the important reasons why the symmetric scheme is used in transport ticketing system other than the asymmetric scheme.

2.5 Background Summary

This background section has explained some technologies related with transport ticketing system, in which RFID is the core technology, and also the core potential security risks in transport system as well. RFID uses challenge response authentication to authenticate and communicate between tags and readers. And also given the security risks and security requirements, they are useful to analyze the security of Oyster card system and give recommendation in the following sections.

Furthermore, system security consideration that we give is a kind of criterion in giving recommendations to improve the system security.

Also, different types of attacks and threats of smart cards and the whole system are explained in details, which give a standardized risk analysis model to help us analyze the security risks in Oyster card system.

In addition, some cryptographic background knowledge is given. With these background technologies and knowledge above, in the following section we will talk about Oyster card system used in London underground as a ticketing medium, and analyze its security.

3 MIFARE CLASSIC

MIFARE is a product family from NXP Semiconductors, which has been widely used as a system for transport industry. Currently MIFARE consists of four different types of cards, Classic, DESFire, Plus and Ultralight. The report from NXP in 2008 shows that more than 1 billion MIFARE cards have been sold and there are about 200 million MIFARE Classic tags being using around the world. Many organizations, including governments, use MIFARE technology as a secure entry system for office buildings, public transport ticketing systems and other applications, especially MIFARE Classic. According to The Times in 2008, about 10 million MIFARE Classic smart cards are sold in Britain each year, providing access to public buildings as well as cashless payment systems for transport.

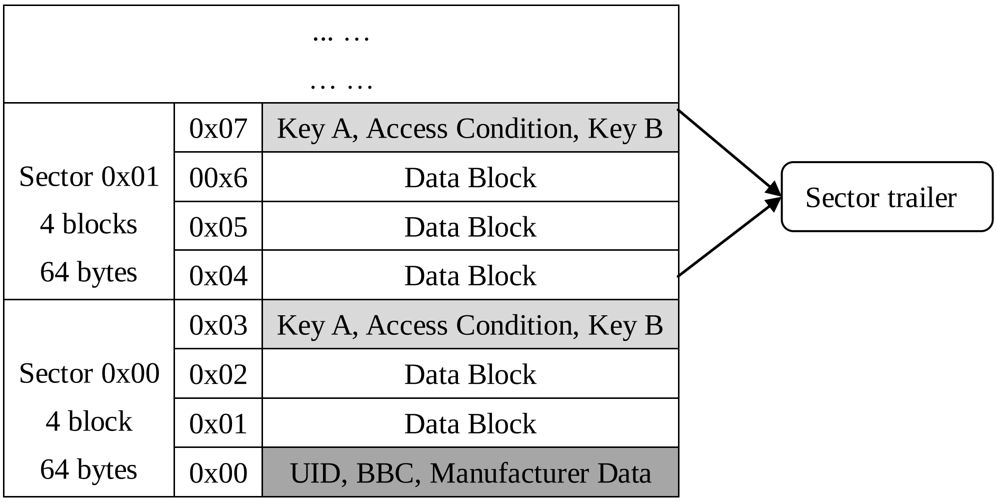
Oyster card, as a ticketing medium used in London Underground, was based on MIFARE Classic as well. However, from February 2011, it is no longer being issued to the public because its security has been proved weak. In March 2008 a group of colleagues at Radboud University in Holland made public that it was able to manipulate the contents of a MIFARE Classic card and clone a new card [7, 16].

In this section, we will talk about the details of MIFARE Classic from logical structure, communication principle, authentication protocol and its encryption algorithm called CRYPTO-1 cipher. Based on the knowledge above, then we will analyze the security risks and show the weaknesses of MIFARE Classic.

3.1 Structures and Protocols

3.1.1 Logical Structure

The MIFARE Classic tag is essentially an EEPROM memory chip with functionality. Basic operations like read block, write block, increment and decrement can be performed on this memory. The memory is divided into data blocks of 16 bytes and these data blocks are grouped into sectors. Every last data block of a sector is called *sector trailer*, which contains two secret keys and access conditions corresponding to that sector. The other blocks of a sector are used to store arbitrary data. Figure 3.1 shows a schematic of the memory of a MIFARE Classic tag.



... … … …		
Sector 0x01 4 blocks 64 bytes	0x07	Key A, Access Condition, Key B
	00x6	Data Block
	0x05	Data Block
	0x04	Data Block
Sector 0x00 4 block 64 bytes	0x03	Key A, Access Condition, Key B
	0x02	Data Block
	0x01	Data Block
	0x00	UID, BBC, Manufacturer Data

Figure 3.1. Memory structure of MIFARE Classical [7].

For each sector trailer, as shown in Figure 3.2, the two secret keys are used for authentication and the access conditions determine which operations are available for that sector. To perform an operation on a specific block, the reader needs to authenticate for the sector containing that block.

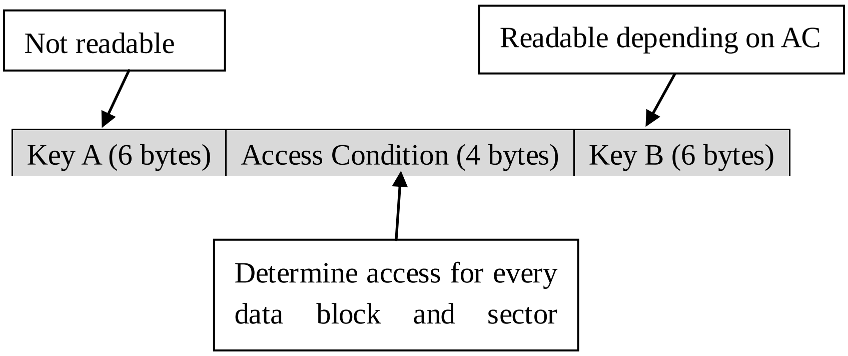

Not readable
Readable depending on AC
Key A (6 bytes)
Access Condition (4 bytes)
Key B (6 bytes)
Determine access for every data block and sector

Figure 3.2. Illustration of sector trailer.

3.1.2 Communication Principle

The communication layer of the MIFARE Classic is based on the ISO/IEC 14443 standard. And the data communications between tags and readers are encrypted and modulated in type A.

Figure 3.3 is a typical communication principle, i.e. transaction procedure, between tags and readers. And we describe the procedure as follows.

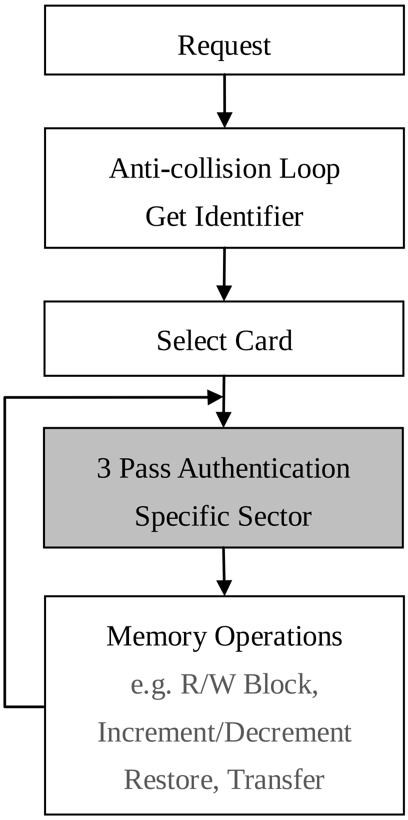


Figure 3.3. A typical transaction procedure between tags and readers.

First of all, the tag answers to a request from the reader after Power-On-Reset.

And then in the anticollision loop, the identifier of the tag is read. If there are several tags in the operating field of the reader, they can be distinguished by their identifier and the right tag is selected for the future transaction. The unselected tags are returned and waiting for new request commands.

After selecting of a tag, the reader specifies the memory location to be accessed and uses the corresponding key for the three pass authentication procedure.

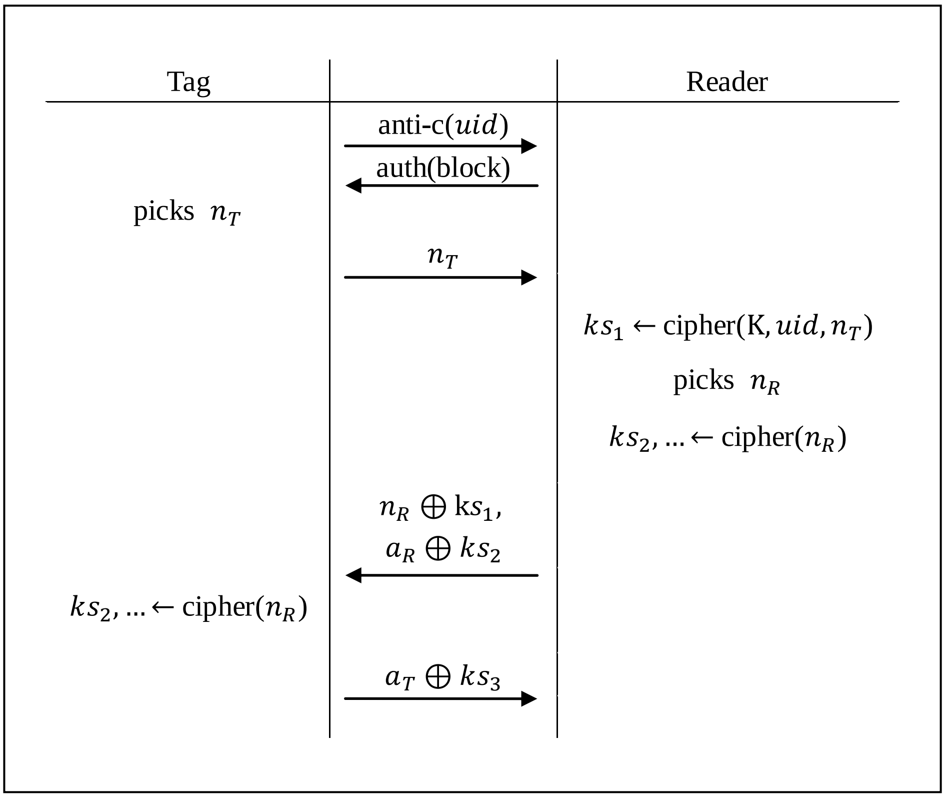
After a successful authentication, the following memory operations can be performed: read block, write block, decrement, increment, restore, transfer.

The most important step related with security in the transaction procedure is the three pass authentication between tags and readers. We will provide details about three pass authentication in the following part.

3.1.3 Authentication Protocol

Since Oyster card is a kind of contactless smart card, it uses RFID technology to communicate with the terminals. As we talked about in Section 2.1.3, one kind of security technology used in the RFID system is challenge response authentication protocol, which is also used in MIFARE Classic. The security of a RFID system relies on the security of its authentication protocol.

According to the manufacture's documentation, we describe the process of the three pass authentication as follows (also see Figure 3.4).



Tag
Reader
anti-c(uid)
auth(block)
picks n_T
n_T
$ks_1 \leftarrow$ cipher(K, uid, n_T)
picks n_R
$ks_2, \ldots \leftarrow$ cipher(n_R)
$n_R \oplus$ ks$_1$,
$a_R \oplus ks_2$
$ks_2, \ldots \leftarrow$ cipher(n_R)
$a_T \oplus ks_3$

Figure 3.4. Three pass authentication protocol of MIFARE Classic.

Let's have R for the reader and T for the tag.

1. R specifies one certain block to be access and chooses key A or key B after

confirmed the identifier of the tag uid.

2. T reads the secret key K and the access conditions from the sector trailer. Then T picks a challenge nonce n_T and sends it to R.

3. R calculates the response a_R using the secret key and additional input. The response a_R, together with a challenge nonce n_R from R, is sent to T.

4. T verifies a_R by comparing it with its own challenge, calculates the response a_T and sends it to R.

5. R verifies a_T by comparing it with its own challenge.

From the protocol, we can see that starting from n_R, n_R, a_R, a_T are all XOR-ed with corresponding parts of the keystream $\mathrm{k}s_1, \mathrm{k}s_2, \mathrm{k}s_3$, which means all communications are encrypted.

Also, we have

$$a_R = \mathrm{su}c^2(n_T)$$

and

$$a_T = \mathrm{su}c^3(n_T)$$

which indicate that a_R and a_T only depend on n_T. If n_T remains constant, then a_R and a_T remain constant as well.

Figure 3.5 shows an example of authentication trace of a tag.

Step	Sender	Hex Bytes	Abstract	
01	Reader	26	req type A	
02	Tag	04 00	answer req	
03	Reader	93 20	select	Anticollision
04	Tag	c2 a8 2d f4 b3	uid, bcc	
05	Reader	93 79 c2 a8 2d f4 b3 ba a3	select(uid)	
06	Tag	08 b6 dd	MIFARE	
07	Reader	60 30 76 4a	auth(block 30)	Authentication
08	Tag	42 97 c0 a4	n_T	
09	Reader	7d db 9b 83 67 eb 5d 83	$n_R \oplus \mathrm{k}s_1, a_R \oplus ks_2$	
10	Tag	8b d4 10 08	$a_T \oplus ks_3$	

Figure 3.5. An example of authentication trace. Source: Ref. [7].

Experiments [7] running several authentication sessions with various uids and tag nonces show that if

$$\mathrm{uid} \oplus \mathrm{n_T}$$

remains constant, the ciphertext of the encrypted reader nonce

$$\mathrm{n_R} \oplus k_{S_1}$$

also remains constant, which implies that the keystream remains constant as well.

3.1.4 CRYPTO-1 Cipher

The cryptographic algorithm that MIFARE Classical used to encrypt the communication between tags and readers is called *CRYPTO-1 cipher*, which is a stream cipher using a 48-bit secret key. The CRYPTO-1 cipher consists of a linear feedback shift register (LFSR) and a filter function, f(·), as shown in Figure 3.6.

During the initialization, the secret 48-bit key is loaded into LFSR and $\mathrm{uid} \oplus \mathrm{n_T}$ is shifted into the state. $\mathrm{n_T}$ is also sent to the reader as a first challenge in a challenge-response protocol in which tags and readers prove knowledge of the secret key.

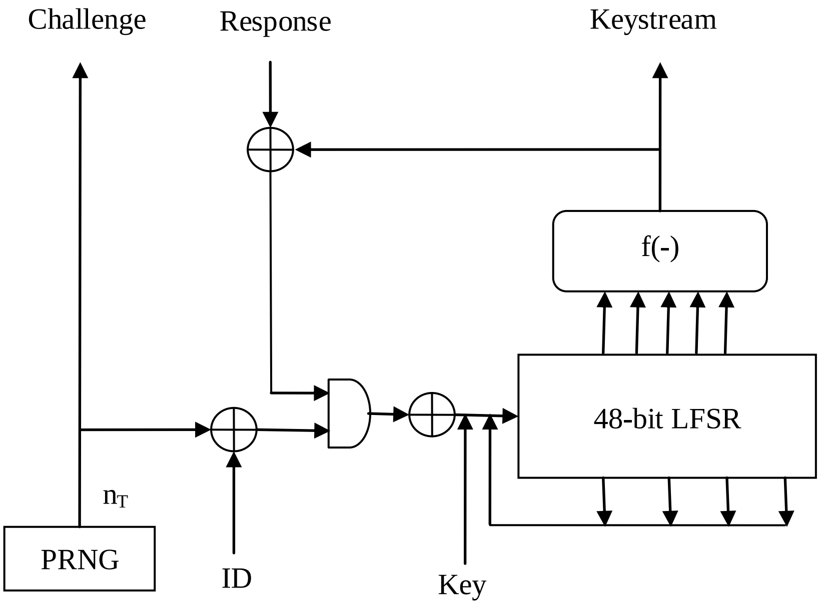


Figure 3.6. CRYPTO-1 cipher and initialization. Source: Ref. [9].

Actually, in this project we do not need to know the detailed knowledge about the CRYPTO-1 algorithm other than that it is just a stream cipher with 48-bit secret key to encrypt bitwise.

3.2 Security Analysis and Weakness

After shows the details about structures and protocols of MIFARE Classic above, we will analyze the security risks and show the weaknesses in MIFARE Classic in the following part. Even without knowing the cryptographic algorithm and the secret key, the keystream can be recovered and then it is able to read and modify card contents. Moreover, the encryption algorithm has been recovered and the secret key has been retrieved as well.

3.2.1 Recover Keystream

It is able to recover keystream without having knowledge about the algorithm and the secret key, what we need is one transaction between a tag and a reader, and completely controlling the reader.

In Section 3.1.3 above, we showed the authentication protocol between tags and readers, in which the tag nonce n_T is generated by PRNG in the tag. However, PRNG uses deterministic processes to generate a series of outputs from an initial seed state, which means, PRNG is fully deterministic [14].

During the experiment [7] by a group of colleagues at Radboud University, a hardware called Proxmark III is used to eavesdrop a transaction. The experiment shows that 600,000 nonces could be requested in one hour and a nonce could be reappeared at least about four times. The nonce is generated by a LFSR which shifts every 9.44µs. Therefore a random nonce could theoretically be reappeared after 0.618s if the tag is queried at exactly the right time. Future more, by fixing n_T and uid, and authenticating repeatedly, the reader produces the same sequence of nonces every time it is restarted.

Also, in the presentation [22] on the 24th congress of the chaos computer club, Nohl and Plotz put forward that the nonce that PRNG generates only depend on the time between power and the start of communication. If we completely control the reader, we control this timing and therefore can get the same tag nonce every time.

So, due to the weakness of PRNG, we can have the same n_T.

Since we have

$$ks_1 \leftarrow \text{cipher}(\mathrm{K}, uid, n_T)$$

If $\mathrm{uid} \oplus \mathrm{n_T}$ remains constant, ks_1 remains constant.

In addition, in Section 3.3.1, we also show that if $\mathrm{uid} \oplus \mathrm{n_T}$ remains constant, the ciphertext of the encrypted reader nonce $\mathrm{n_R} \oplus k_{S_1}$ also remains constant.

Since we have

$$ks_2, \ldots \leftarrow \text{cipher}(n_R)$$

If $\mathrm{n_R} \oplus k_{S_1}$ remains constant, ks_2 remains constant.

Therefore, due to the weakness of PRNG, we can get the same tag nonce and have the same keystream if we completely control the reader.

Furthermore, it is easy to see that parts of keystream can be recovered if the adversary eavesdrop one transaction, n_T and $a_R \oplus ks_2, a_T \oplus ks_3$.

Since we know

$$a_R = \mathrm{su}c^2(n_T)$$

and

$$a_T = \mathrm{suc}^3(n_T)$$

by computing $\mathrm{suc}^2(n_T), a_T = \mathrm{suc}^3(n_T)$ and XOR operation, we can calculate ks_2, ks_3, etc.

Replay attack

Since it is able to recover the same keystream, then it is quite possible to replay an earlier recorded transaction between a tag and a reader, and then read and modify card contents without knowing the secret key.

For example, the plaintext P_1, P_2 in the communication are XOR-ed bitwise with the same keystream K which gives the encrypted data C_1 and C_2.

$$\mathrm{P}_1 \oplus \mathrm{K} = \mathrm{C}_1$$
$$\mathrm{P}_2 \oplus \mathrm{K} = \mathrm{C}_2$$

When it is possible to use the same keystream on a different plaintext P_2, and either P_1 or P_2 is known, then both P_1 and P_2 are revealed.

$$\mathrm{C}_1 \oplus \mathrm{C}_2 \Rightarrow \mathrm{P}_1 \oplus \mathrm{P}_2 \oplus \mathrm{K} \oplus \mathrm{K} \Rightarrow \mathrm{P}_1 \oplus \mathrm{P}_2$$

The following describes the steps.

1. Eavesdrop one transaction between a tag and a reader.

2. Control the reader to make sure the tag uses the same keystream as in the recorded communication.

3. Modify the plaintext, for example, changing the block number in a read block command, such that the tag receives a command for which we know plaintext in the response.

4. Compute the corresponding keystream segment for each segment of known plaintext.

5. Use this keystream to partially decrypt the trace obtained in step 1.

6. Recover more keystream bits by shifting commands.

Based on this principle, not only replay an earlier recorded transaction, but we can flip ciphertext bits to try to modify the first command such that it gives another result. In that way, we can read and modify card contents without knowing the secret key, what we need is just transactions between a tag and a reader.

3.2.2 Retrieve Secret Key

As we mentioned in Section 3.1.4, MIFARE Classic tag provides mutual authentication and data secrecy by means of CRYPTO-1 cipher. The obvious and fundamental limitations of CRYPTO-1 have long been well-known in the expert community.

- The algorithm uses a very short key, 48 bits
- The security of the card strongly relies on the secrecy of the algorithm

In 2008, CRYPTO-1 algorithm had been recovered by a group of colleagues at Radbound University in Holland and the secret key had been retrieved from a MIFARE reader as well.

The CRYPTO-1 cipher consists of a linear feedback shift register (LFSR) and a filter function, f(·). Garcia, et al. [7] succeeded in inverting the filter function using two ways, and also with a LFSR rollback technique, they recovered the secret key.

After revealed the secret key and recovered all the information on the tag, a new card can be cloned and fraudulent activities could be happened, for example, the cloned card owner could travel without paying by himself or herself, and the electric money on the genuine card could be stolen.

Basically, the ability to retrieve the secret key is closely related to the length of the key. When MIFARE Classic was introduced over a decade ago, keys were no longer than 48 bits, which makes a brute force attack faster and easier using recent technologies in today’s world. According to Kerckhoffs’ principle, the security of a cryptographic system should reside in secrecy of the key, not in secrecy of its algorithm or the system itself.

3.3 Summary

MIFARE Classic, based Oyster card, uses three pass authentication to encrypt the communication between cards and readers, and uses CRYPTO-1 cipher algorithm with 48-bit secret key. The security of this system strongly relies on the three pass authentication and the algorithm.

However, MIFARE Classic has been proved very weak. On one hand, it is able to recover keystream and then read and modify the contents of Oyster cards because of the weaknesses of PRNG and authentication protocol. On the other hand, the encryption algorithm has been recovered and the secret key has been retrieved as well, due to the weak secrecy of algorithm itself and the shortness of key size.

Because of the vulnerability and weakness, MIFARE Classic is no longer being used as a transport ticketing system in London Underground. To solve this problem, some recommendations and measures should be taken to improve the whole security of this system, or propose another system with high level security as an alternative.

4 MIFARE DESFIRE

Since the security of MIFARE Classic is terrible and it is no longer being used in public, as an alternative, from 2009, MIFARE DESFire based Oyster cards is now widely being used as a transport system in London underground.

MIFARE DESFire is another product from NXP Semiconductors. According to the documentation by NXP, the main characteristics of MIFARE DESFire are denoted by its name "DESFire". DES indicates the high level of security using DES, 3DES or AES cryptographic algorithm for enciphering transmission data. Fire indicates its outstanding features, Fast, Innovative, Reliable and Enhanced [23].

Table 4.1 shows the features between MIFARE Classic and MIFARE DESFire. Similar with MIFARE Classic, MIFARE DESFire is based on the open global standards ISO/IEC 14443A for both air interface and cryptographic methods.

Compared with MIFARE Classic, the main differences on security features of MIFARE DESFire are as follows, which shows the higher secure level and more complex computation power.

1. MIFARE DESFire uses the high quality encryption algorithm, DES/ 3DES/ AES, which is much more secure than CRYPTO 1.
2. The key size the algorithm used is over 100 bits, which is much longer than that in MIFARE Classic and reduces the attractiveness of brute force attacks since they take exponential orders of magnitude more time and computing effort.
3. Data encryption on RF-channel with replay attack protection.
4. Authentication on application level.
5. 14 access keys for per application.
6. Automatically backup the data in the back end system.

Table 4.1. Features between MIFARE Classic and MIFARE DESFire [23].

Product Features	MIFARE Classic 1K	MIFARE DESFire 2K
Memory		
Size	1024	2048
Organization	16 sector a 64 byte	flexible file system
Security		
Serial Number (byte)	4Byte NUID or 7Byte UID with optional random ID#	7 Byte UID
PRNG	yes	yes
Access Keys	2 keys per sector	14 keys per application
Access Conditions	per sector	per file
Key Size	48 bits	56 / 112 / 168 bits, 128 bits
Encryption Algorithm	CRYPTO-1	DES / 3DES / AES/ CMAC
Anti-tear supported by chip	for value blocks	YES
Special Features		
Multi-applicatio n	supports MAD*	28 applications, MAD3***
Special Funcationalities	-	Automatical backup mechanism Random ID (optional)

As we concluded the weaknesses of MIFARE Classic in Section 3.2, it is possible to replay an earlier recorded transaction, i.e. replay attack, and recover the keystream by eavesdropping one transaction between a tag and a read. While in MIFARE DESFire, the RFID channel has been encrypted and it is difficult to do a replay attack even if one transaction is eavesdropped and some encrypted data are revealed.

Also, MIFARE DESFire brings many benefits to end users. Cardholders can experience convenient contactless ticketing way while also having the possibility to use the same device for other related applications. MIFARE DESFire does offer enhanced consumer-friendly system design, in combination with security and reliability [23].

However, although the official documentation by NXP shows that the security of MIFARE DESFire is much higher than MIFARE Classic, some facts have shown that MIFARE DESFire chip in Oyster card is already problematic. In an emergency brief issued by ATOC in 2010, it quotes

> *"…since their introduction there have been an increase in reported cases of the new (Desfire) Freedom Pass taking longer to be 'read' by some gate reader software."*

And it goes

> *"As a consequence some Oyster users believe their new cards have failed when gates do not open. Where this occurs, gateline staff should ask the cardholder to re-present their Oyster card and actually 'touch it' on the gate reader. If the reader still rejects the new style Oyster card, the holder should be allowed to travel……"*

And also some reports show that some passengers had being charged a penalty fare of £50 because the hand held readers which inspectors use cannot read the cards.

However, the problem is that since MIFARE DESFire based Oyster cards has being used widely in London underground until now, the technological details about MIFARE DESFire have not been released by NXP. And there are not too many public available documents to descript it.

Although since now we cannot analyze the security of MIFARE DESFire in details, such as authentication protocol, we can still give some recommendations and replacements based on the known weaknesses of MIFARE Classic. Any recommendations or replacements should be based on an algorithm that has been rigorously assessed by the cryptographic expert community.

5 RECOMMENDATION

In this section, we will propose some recommendations to enhance the security of the system used in Oyster card. As we mentioned in Section 2.2, security systems should be designed on the principle of prevention, detection and reaction. So the first recommendation we give is based on prevention, which is to strengthen the security of the authentication protocol between cards and terminals in MIFARE Classic system we talked about in Section 3.1.3. And the second one is on the principle of detection, which is to detect a counterfeit card to be used. In addition, we give some recommendations about protecting personal information secure from perspective of end users.

5.1 Encrypt Authentication Protocol

Both MIFARE Classic system and MIFARE DESFire system use RFID technology to authenticate and communicate between Oyster cards and terminals. And the security of a RFID system relies on the authentication protocol. As showed in Section 3.2, the authentication protocol of MIFARE Classic is weak. If an adversary eavesdrops one communication between a tag and a reader, and completely control the reader, then the system can be attacked. So the first recommendation we give is to enhance the security of authentication and communication between Oyster cards and terminals. Even if an adversary eavesdrops data during the authentication process, the system is still hard to be attacked due to the data is useless and meaningless to the adversary.

5.1.1 Related Work

As we talked about RFID system in Section 2.1.3, the communication between tags and readers is performed at insecure channel through RF channel. To solve this problem, many kinds of security technologies for enhancing the security of the RFID system are proposed and most of them are based on hash function.

One of the methods is the hash-locking protocol [28], which uses *metalID* to hide tag's real ID. *MetalID* is a hash value of a random key, i.e.

$$metalID \leftarrow hash(key)$$

which is stored in tag's memory, together with tag's real ID. Also both the key and the *metalID* are stored in the back end database. To unlock a tag, the reader queries the *metalID* from the tag, looks up the roight key in the back end database and finally transmits the key to the tag. Then the tag hashes the key and compares it to the stored *metalID*. If the two values match, the tag is unlocked and offers its functionality to the readers. This protocol is illustrated in Figure 5.1.

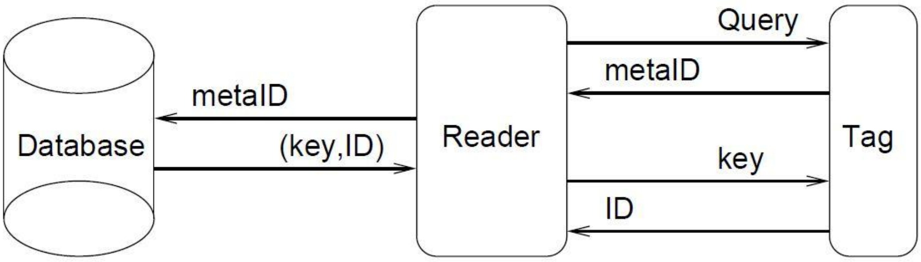


Figure 5.1. The hash-locking protocol. Source: Ref. [28]

Based on the difficulty of inverting a one-way hash function, this scheme prevents unauthorized readers from reading tag contents. However, it is vulnerable to spoofing attack if an adversary disguises the right reader and receives the *metalID* from the tag, and then disguises the right tag and gets the key from the right readers by sending the *metalID* [26].

Another method is using randomized hash-locking protocol [28] by cryptographic approach. In the randomized hash-locking protocol, the tag is equipped with a one-way hash function, and also a PRNG. As shown in Figure 5.2, we describe the protocol as follows.

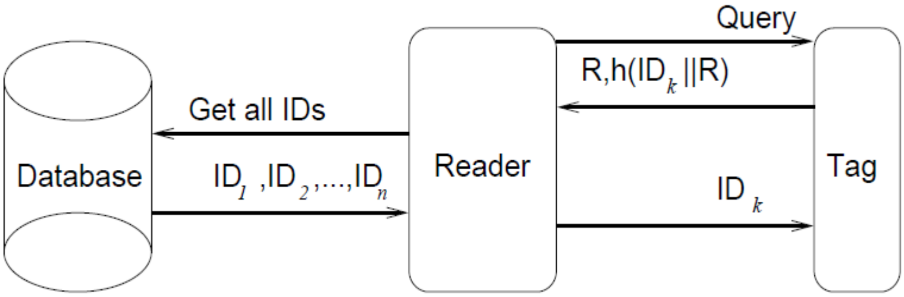


Figure 5.2. The randomized hash-locking protocol. Source: Ref. [28]

1. The reader sends a query to the tag.

2. The tag responds to the query by generating a random value R, then hashing its ID concatenated with R and sending both values to the reader, that is,

$$(\mathrm{R}, h(ID_k \| R))$$

3. The reader then sends the values $(\mathrm{R}, h(ID_k \| R))$ to the database.

4. The database (and the reader) identifies the tag by performing a brute force search of all known IDs and hashing each of them concatenated with until it finds a match.

The advantage of this protocol is that the communication between tags and reader are encrypted. Even if an adversary eavesdrops one communication and knows the values (R, $h(ID_k\|R)$), these numbers are meaningless to the adversary. And it is hard for the adversary to recover the value of id because of the one-way property of hash function.

However, similar to the hash-locking protocol, this mode is vulnerable to spoofing attack since the attacker can disguise the tag. Moreover, it is only feasible for a relatively small number of tags. If the number of tags is huge, e.g. Oyster card, it is impractical to be used due to the time consuming of brute force search.

Beside these cryptographic approaches, some physical technologies for security are proposed, such as Kill Command [28]. However, the physical technologies to protect RFID system are hard to implement, because additional devices are needed or the form of the protection devices is limited. So our proposed protocol in the following part is based on cryptographic technology.

5.1.2 Proposed Protocol

As we analyzed the security of MIFARE Classic system in Section 3, we know that the communicated authentications between tags and reads are weak. If an adversary eavesdrops one communication, then uid and n_T can are revealed and more information on the tag would be revealed.

To improve the authentication security, we propose a method that encrypts uid and n_T using hash function during the communication process. As shown in Figure 5.3, we describe our proposed process of authentication as follows.

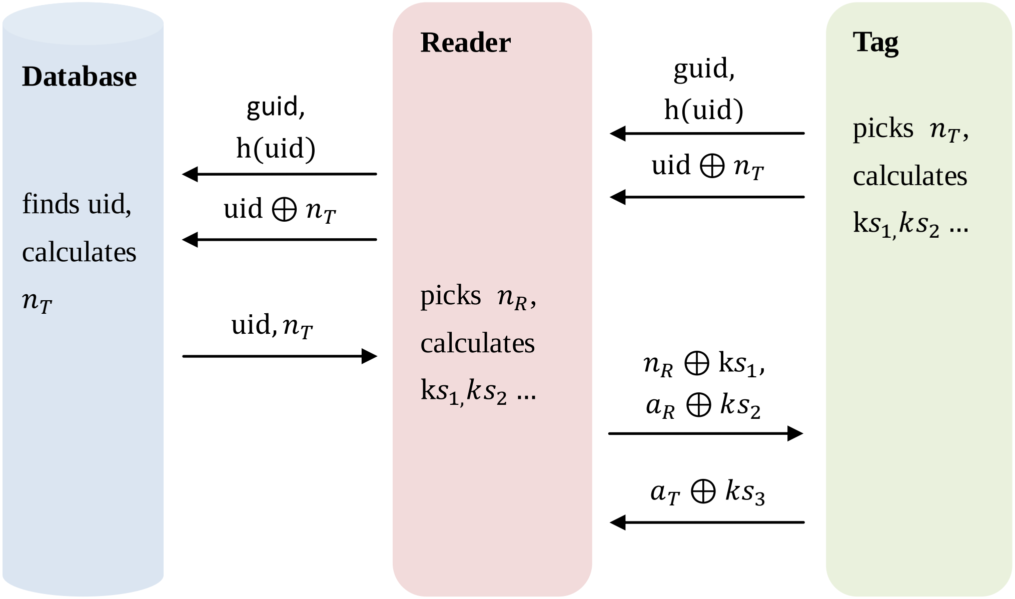

Database
finds uid, calculates n_T
guid, h(uid)
uid $\oplus$ n_T
uid, n_T
Reader
picks n_R, calculates ks_1, ks_2 ...
guid, h(uid)
uid $\oplus$ n_T
$n_R \oplus ks_1$, $a_R \oplus ks_2$
$a_T \oplus ks_3$
Tag
picks n_T, calculates ks_1, ks_2 ...

Figure 5.3. Our proposed authentication protocol.

We have R for the reader, T for the tag and DB for the database.

1. T picks a challenge nonce n_T and sends the encrypted $\text{uid} \oplus n_T$ to R, together with its hashed identifier h(uid).

2. R sends the received $\text{uid} \oplus n_T$ and h(uid) to DB.

3. DB searches all the uids it stored until it finds the right uid of T.

4. DB calculaters n_T, and sends it to R, together with uid.

5. R calculates the response a_R using the secret key and additional input. The responsea_R, together with a challenge nonce n_R from R, is sent to T.

6. T verifies a_R by comparing it with its own challenge, calculates the response a_T and sends it to R.

7. R verifies a_T by comparing it with its own challenge.

In step 3, in order to find the right *uid* of the tag, the database searches all the *uid* it stored by performing a brute force search of all its know *uid* by hashing each of them until it finds a match. It is feasible to use when the application is small as well as the database in the system. However, in the large applications or systems, like Oyster card and its system, the database is huge and it is impracticable to use a brute force search to find the right tag, which would take a long time.

To solve that problem, we propose that dividing all the *uid* into several groups named *guid*. Both *uid* and *guid* are stored in the memory of the tag as well as the database, as shown in Figure 5.4.

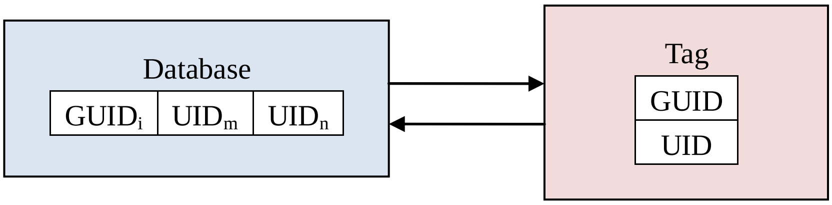


Figure 5.4. Illustration of data stored in tags and database.

Therefore in step 1, the tag also sends its *guid* to the reader, and the reader sends it to the database as well. When the database searches the tag's *uid*, it searches its *guid* firstly. In that way, the time consuming of a brute force search has been reduced and the efficiency of searching has been improved, which depends on the size of *guid*.

5.1.3 Security Analysis

Since the authentication protocol is based on RFID technology, we analyze the security of our proposed protocol based on RFID security requirements. And our results are concluded by comparing with the security of the original protocol. Just like the general RFID systems, we assume the communication channel between readers and back-end databases is secure.

Data confidentiality

By utilizing the hash function, the information communicated between tags and readers are encrypted, especially, uid and n_T. Even if an adversary eavesdrops the uid $\oplus n_T$ and h(uid), due to the one-way property of hash function, it is hard to get the value of uid and n_T. So the private information of the tag is kept secure to guarantee users' privacy.

Indistinguishability

If an adversary eavesdrops the data during one authentication, because n_T and n_R are random values, they are useless to trace the tag. Even due to the weakness of PRNG, the adversary still cannot recover any values because of the one-way property of hash function. So responses emitted by a tag cannot be discriminated from others.

Forward security

In our protocol, it is hard or even not possible to trace the data back through previous events in which the tag was involved. The adversary cannot associate the current output with previous output if he or she only eavesdrops the tag's output. Even if an adversary knows the current security key and has eavesdropped plenty of communications, the forward trace is still impossible over successive authentication.

5.2 Detect Cloning Attack with Synchronized Secrets

As we talked about RFID technology in Section 2.1.3, one of the most challenging security threats in commercial RFID applications is tag cloning. Many researches address these threats primarily by trying to make tag cloning hard by using cryptographic tag authentication protocols [13]. Actually it is not very difficult to protect a tag from cloning using that way in today's world, but it is extremely challenging to implement because of the RFID industry's desire to limit tag hardware functionality in order to produce low-cost tags. Passive tags are limited by the amount of power they can obtain from RFID readers [25]. As the transaction time between tags and readers is limited, in order to supply cryptographic components with sufficient power, tags need to be read from a shorter distance, which degrades the read-rate of RFID readers [25].

Detective measures do not required cryptographic operations based on the tags, but they make use of visibility to detect cloned tags or changes in the tag ownership. The efficiency of detective measure is characterized by the probability of detecting a threat [18]. Detective measures can generate false alarms for example, to prove the detecting cloning attack happened when a genuine tag is classified as a cloning.

This recommendation we propose is base on detective measure and the underlying technical concept is simple and easy for implementation.

5.2.1 Previous Work

Many researches have been focused on detecting cloned tags over decades. For example, Mirowski and Hartnett [20] developed a system called Deckard, that could essentially detect cloned RFID tags attack or other changes of tag ownership in an access control application using intrusion detection methods. The method is inspired by the Intrusion Detection Expert System (IDES) by Denning [6]. In this system, a tag' audit record is used to build a profile of normal behavior, which can be used to determine when subject behavior significantly deviates. A significant deviation away from normal behavior is an indication of a change of tag ownership. This system is useful in detecting when an attacker starts using a cloned or stolen tag.

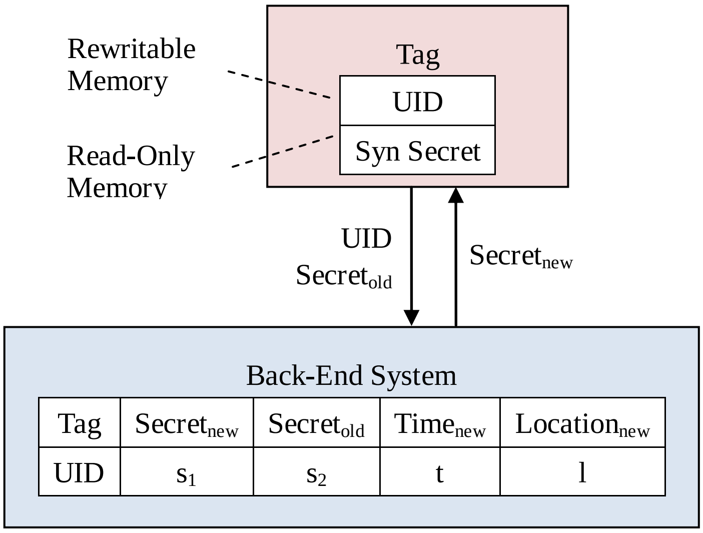
Also, Ilic et al. [10] proposed a kind of synchronized secret approach, but the application focused on the ownership transfer and access control. Grummt and Ackermann [8] put forward a similar idea based synchronized secrets approach in a RFID access controlled application in a scheme called chosen temporarily valid secrets. In addition, Koscher et al. [17] described the same principle to enhance the security of access code based authentication of EPC tags.

However, those proposed methods or systems are focus on transferring the ownership of tags and authentication of the tags. Few of them have discussed and evaluated how the synchronized secrets approaches are applied to detect tag cloning attacks, which our proposed method will focus on.

5.2.2 Proposed Method

This proposed method is base on detective measures and makes use of EEPROM of the tag. Not only the uid of the tag is stored in the read-only memory, but a random number, we called *synchronized secret*, is stored in tag's rewritable memory. The random number is changed every time when the tag is read. Since it is unknown and secret to all who do not have access to the tag, we can take it as a one-time password. A centralized bank-end system issues and keeps tracks of these numbers.

Figure 5.5 shows the overall principle of detecting tag cloning system.



Tag	$\text{Secret}_{\text{new}}$	$\text{Secret}_{\text{old}}$	Time_{new}	$\text{Location}_{\text{new}}$
UID	s_1	s_2	t	l

Figure 5.5. The overall principle of detecting tag cloning system.

We describe the authentication protocol of the proposed detecting tag cloning system as following, as shown in Figure 5.6. In the illustration,

- s_i denotes the current synchronized secret and s_{i+1} is the new one.
- PRNG is used to generate a random number as a synchronized secret.
- ACK denotes an acknowledgement of a successful update of synchronized secrets.

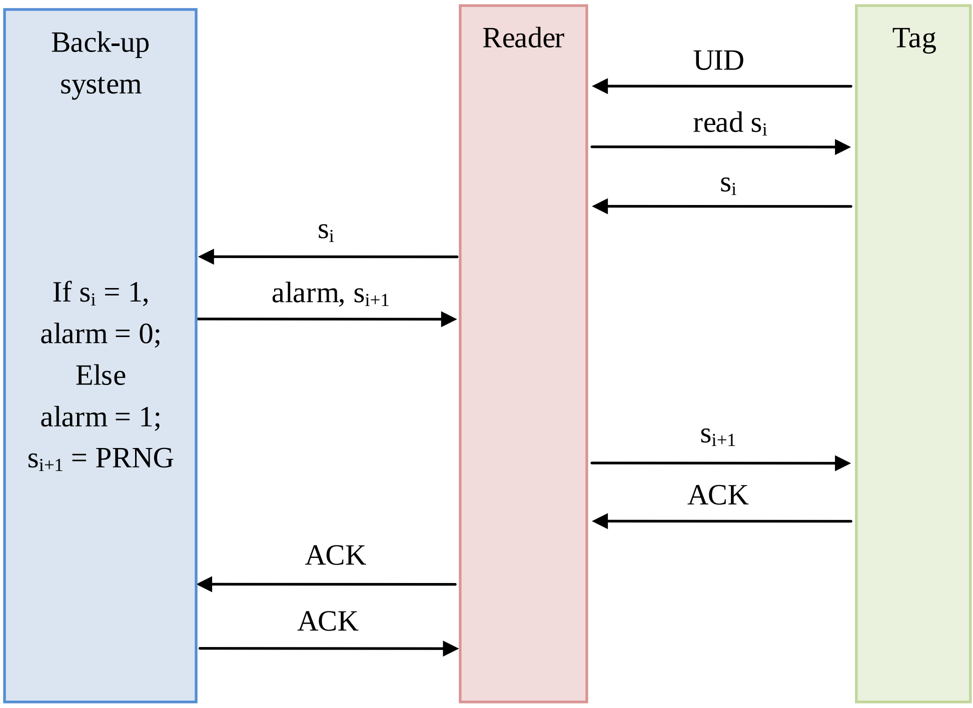


Figure 5.6. The authentication protocol of detecting tag cloning system.

1. The back-end database first verifies the identifier of the tag when the tag is read every time.
2. If the identifier is valid, the back-end system compares the tag’s synchronized secret with the one it stored for that certain tag.
3. If the two numbers match, then the tag passes the check, otherwise a false alarm is triggered.
4. After the check, the back-end system generates a new synchronized secret written on the tag.

In this method, an outdated synchronized secret is used as a strong evidence of a tag cloning attack, however, it along does not prove that that tag is cloned. For example, if the cloned tag is read before the genuine tag after cloning attack occurred, it is the genuine tag that has an outdated synchronized secret and being treated as a cloned tag. Also, if a tag has a valid identifier but a synchronized secret has never been issued by the back-up system or the issuing time is delayed, the tag is likely to be forged.

Therefore, this method manages to detect the tags with the same identifier but cannot tell which one is genuine and which one is cloned. To make a distinction between a genuine tag and a cloned tag, it is still necessary to use a manual inspection, for example, checking the future information stored in the tag or back-end database.

In our method, not only the old and new synchronized secrets are stored in back-end database, but the current location and current time when the tag is scanned are also stored. If a cloning attack has occurred, the back-end system will alert a window shown the current time and location when and where the attack occurred. After checking the data using a manual inspection, it is easy to detect the cloned tag.

5.2.3 Evaluate Security

The level of security of a detection based security measure is characterized by its detection rate [18]. We give a scenario and find that three mutually exclusive outcomes are possible when a cloning attack occurs, as shown in Figure 5.7. In the illustration,

- T_{update} is a random variable as the time between updates for a tag.
- T_{attack} is also a random variable as the time delay from when the cloning attack occurs to when the cloned tag is scanned.

We describe the three outcomes as following.

Case 1: The cloned tag is scanned after the genuine tag, thus a false alarm is triggered when the cloned tag is scanned.

Case 2: The cloned tag is scanned before the genuine tag, thus a false alarm is triggered when the genuine tag is scanned.

Case 3: The genuine tag is not scanned anymore, thus no false alarm is triggered when the cloned tag is scanned.

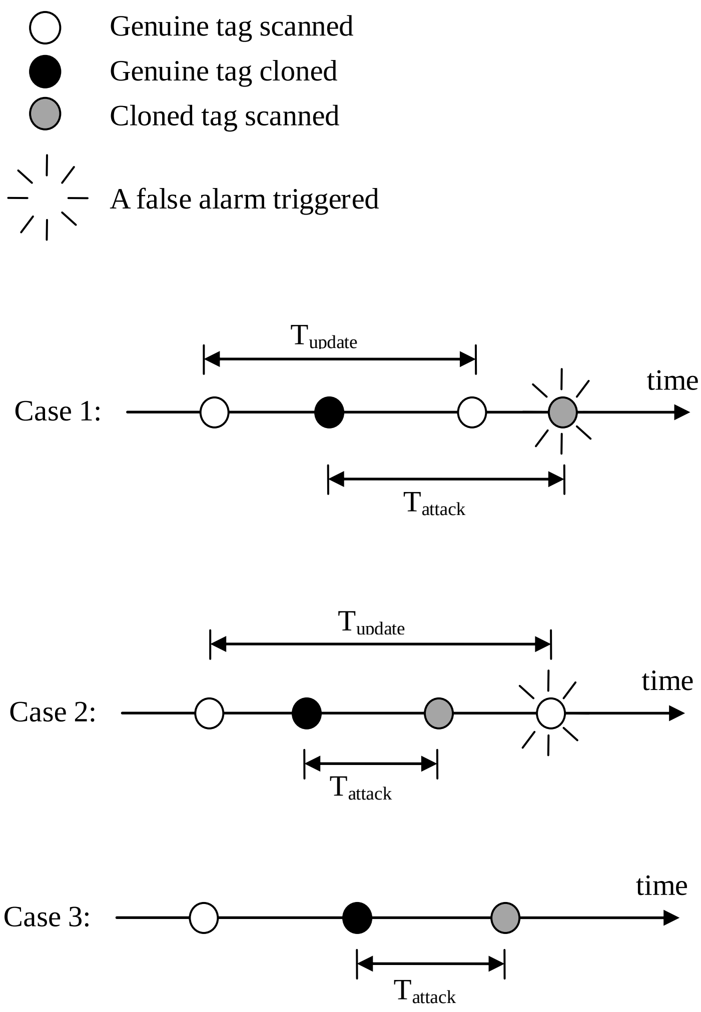


Figure 5.7. Three possible outcomes of a cloning attack [18].

In case 1, when the cloned tag is scanned for the first time, a false alarm is triggered and the cloned tag is detected, thus the negative effective can be prevented. In case 2, when the cloned tag is scanned for the first time and before the genuine tag, no false alarm is trigged and the cloned tag passes the check. While, when the genuine tag is scanned for the first time after it was coned, the system will detect the cloning attack and false alarm is trigged. However, since the cloned tag passes the check for the first time when it is scanned, some negative effectives are still happened. In case 3, the genuine tag is not scanned anymore, thus the cloned tag can be as a genuine tag and the detecting system totally fails.

From the detailed illustration above, we can find that the probability of case 1 equals the probability of the genuine tag is scanned before the cloned tag, multiplied by the probability that the genuine tag is scanned at least once more. The probability of case 1 shows how often the cloning attacking can be prevented. And the probability of case 1 or case 2 equals the probability of genuine tag is scanned at least once more, and it shows how often the cloning attacking can be detected.

So, we conclude that the level of security of our proposed detecting cloning attack with synchronized secrets is characterized by the probability of case 1 or case 2, in other words, depends on the probability that the genuine tag is scanned once more and on the frequency of that the genuine tags are scanned related to the time delay of the cloning attacking.

In the real world, the probability of case 3, that is, a cloned tag always goes unnoticed, is quite small. Some measure can be taken to minimize the probability of case 3 from the individual's personal responsibilities, which we will talk about in the following section.

5.3 From Perspective of Users

It is not only the responsibilities of system designers, but also the end users to take some measures to keep individual’s personal information secure. For example, shorten the time of communication between the card and the terminal in case an adversary has enough time to eavesdrop the communication using certain equipment.

Also, it is important that only using the right secure ways that the official company provides to purchase, top up or transaction the cards. Some people may ask for your Oyster card for a variety of reasons but make sure that you only give your card to the trusted sources. This will minimize the potential risks that the valuable information on the cards is stolen.

Even if the card is lost or stolen, take action immediately with the authenticated parties. When alerted, the issuers will carry out proper measures to protect your privacy, for example lock the cards and render them useless. None of the private data will be lost and accessed or in worse situations exploited resulting huge personal losses. In addition, when a smart card is lost or stolen, the information stored on it will not be lost thanks to the backup system. In fact, smart cards are just affiliated devices to store users’ data. The main data are stored in the back end databases of issuers’ systems [21].

Moreover, if the card is no more used, cancel the card thus the personal information stored in the back end databases will be deleted.

5.4 Summary

Since both Oyster cards and back end databases store lots of highly valuable individual information, such as name, data of birth, credit account and biometric data, it is necessary and responsible for both security designers and end users to take some measures to enhance the security of system. Two recommendations are given from the perspective of security designers. The first one is proposed on the principle of prevention by using hash function to enhance the security level of the authentication protocol such that the communications between cards and terminals are encrypted. And the second one is based on detection by using synchronized secrets to detect card cloning attacks. Also, we gave some advices about protecting personal information secure from individual's personal responsibilities.

6 CONCLUSION AND FURTHER WORK

6.1 Conclusion

The prevailing usage of smart cards lie in their convenience for communication and authentication, their potentially strong security for data storage, and their facilitation for multiple accessibility and definitive audit trails. They embody the spirits of trust, privacy, and of course convenience. They offer a clear advantage to the diverse parties involved: issuers, merchants, and customers. Smart cards often contain highly valuable pieces of information, such as credit account, identification, and biometric data. The importance of the functions of smart cards render security, particularly privacy protection, imperative.

MIFARE Classic, based Oyster card, was used as a transport ticketing system in London underground. It has been proved that it is able to recover keystream and then read and modify the contents of Oyster cards because of the weaknesses of PRNG and authentication protocol. Moreover, the encryption algorithm has been recovered and the secret key has been retrieve as well, due to the weak secrecy of algorithm itself and the shortness of key size. As an alternative, MIFARE DESFire is now widely being used, which has higher security level than MIFARE Classic.

Bases on the analysis of Oyster card system, some recommendations have been proposed in this project from the perspective of security designer to enhance the security. By using hash function to enhance the security of the authentication protocol, the data that a card and a reader communicated with each other are encrypted. It is hard to decrypt the data because of one-way property of hash function. The second recommendation is to use synchronized secrets such that card cloning attack can be detected. The synchronized secrets are stored in both the memory of cards and back end database, when a card is scanned, the database check the synchronized secret that transmitted by the card by comparing with the one it stored. If the two synchronized secrets do not match, the card cloning attacks happen. In addition, some recommendations are given from the responsibility of end users.

6.2 Further Work

Substantial efforts have been exerted by academic professionals and industrial practitioners to protect the data privacy of smart cards. Our attempt in this paper is to add extra and hopefully stimulating knowledge into this area. The two recommendations we gave are theoretical proved to enhance the security level of Oyster card system. Apart from this, other important consideration in designing a system are to understand the real commercial values of gaining from both the side of the developers and the attackers.

In other words, the overall cost by the utilized technology should be considered. Also, considerations of feasibility and implementation should be taken as well. If the security is in a high level but the costs and efforts on this system is very high, at the same time, the system is not widely used and the commercial gaining for the related entities is low, then the proposed method or technology is not feasibility. Furthermore, the costs and efforts from the side of attackers also should be considered as critical factors to characterize the feasibility and security of the system.

Thus it is necessary to well balance the overall advantages and disadvantages of smart card technology in the public transport system when pursuing feasibility, interoperability and security, which means, to well balance the costs, efforts and future benefits from the perspectives of security designers and attackers.

7 REFERENCES

[1] Blum, L., Blum, M. and Shub, M. (1986). A simple unpredictable pseudo-random number generator. *SIAM Journal on Computing*. 15(2): 364-383.

[2] Blum, M. and Micali, S. (1984). How to generate cryptographically strong sequences of pseudo-random bits. *SIAM Journal on Computing*. 13(4): 112-117.

[3] Blythe, P.T. (2004). Improving public transport ticketing through smart cards. *Proceedings of the ICE - Municipal Engineer*. 157(1): 47-54.

[4] Cavoukian A. (2009). Privacy by Design. *The Information & Privacy Commissioner of Ontario*. http://www.ipc.on.ca/english/Privacy/Introduction-to-PbD/

[5] Choi, E.Y., Lee, S.M. and Lee, D.H. (2005). Efficient RFID authentication protocol for ubiquitous computing environment. *Embedded and Ubiquitous Computing – EU 2005 Workshops*. LNCS. 3823: 945-954.

[6] Denning, D.E. (1987). An intrusion-detection model. *IEEE Transaction on Software Engineering*. 13(2): 222-232.

[7] Garcia, F.D., Koning Gans, G., Muijrers, R., Rossum, P., Verdult, R., Schreur, R.W., Jacobs, B. (2008). Dismantling MIFARE Classic. *Computer Security – Esorics 2008*. LNCS 5283: 97-114.

[8] Grummt, E. and Ackermann, R. (2008). Proof of possession: Using RFID for large-scale authorization management. *Communications in Computer and Information Science*. 11(4): 174–182.

[9] Hohl, K. (). Cryptanalysis of Crypto-1. University of Virginia. http://www.cs.virginia.edu/~kn5f/Mifare.Cryptanalysis.htm

[10] Ilic, A., Michahelles, F., Fleisch, E., Manage, I. and ETH Zurich, Z. (2007). The dual ownership model: using organizational relationships for access control in safety supply chains. *AINAW '07 Proceedings of the 21st International Conference on Advanced Information Networking and Applications Workshops*. pp. 459-466.

[11] Information and Privacy Commissioner of Ontario. (2010). Privacy Risk Management: Building privacy protection into a Risk Management Framework to ensure that privacy risks are managed, by default. http://www.ipc.on.ca/images/Resources/pbd-priv-risk-mgmt.pdf

[12] ISO/IEC 14443. (2001). Identification cards - contactless integrated circuit cards proximity cards. *ISO's/IEC's Joint Technical Committee 1*.

[13] Juels, A. (2006). RFID security and privacy: a research survey. *IEEE Journal of Selected Areas of Communication*. 24(2): 381-394.

[14] Jun, B. and Kocher, P. (1999). The intel random number generator. *Cryptography Research, Inc. and Intel Corporation*.

[15] Kocher, P., Jaffe, J. and Jun, B. (1999). Differential power analysis. *Advances in Cryptology – CRYPTO ' 99*. LNCS. 1666: 388-397.

[16] Koning Gans, G., Hoepman, J-H., Garcia, F.D. (2008). A practical attack on the MIFARE Classic. *Smart Card Research and Advanced Applications*. LNCS 5189: 267-282.

[17] Koscher, K., Juels, A., Brajkovic, V. and Kohno, T. (2009). EPC RFID tag security weaknesses and defenses: passport cards, enhanced drivers licenses, and beyond. *CCS '09: Proceedings of the 16th ACM conference on Computer and communications security*. pp. 33-42.

[18] Lehtonen, M., Ostojic, D., Ilic, A. and Michahelles, F. (2009). Securing RFID systems by detecting tag cloning. *Pervasive Computing*. LNCS 5538: 291-308.

[19] Markantonakis, K., Mayes. K., Sauveron. D. and Askoxylakis. I. G. (2008). Overview of security threats for smart cards in the public transport industry. *Proceedings of the 2008 IEEE International Conference on e-Business Engineering*. pp. 506-513.

[20] Mirowski, L. and Hartnett, J. (2007). Deckard: a system to detect change of RFID tag ownership. *International Journal of Computer Science and Network Security*. 7(7).

[21] Neame, R. (1997). Smart cards-the key to trustworthy health information system. *BMJ*. 314: 573-577.

[22] Nohl, K. and Plotz, H. (2007). MIFARE, little security, despite obscurity. *Presentation on the 24th Congress of the Chaos Computer Club*. Berlin.

[23] NXP Semiconductors. (2010). MIFAE DESFire EV1 contactless multi-application IC. http://www.nxp.com

[24] Ohkubo, M., Suzuki, K. and Kinoshita, S. (2003). Cryptographic approach to “privacy-friendly” tags. *RFID Privacy Workshop 2003, MIT.*

[25] Ranasinghe, D.C., Engels, D.W. and Cole, P.H. (2005). Low-cost RFID systems: confronting security and privacy. In: *Auto-ID Labs Research Workshop*.

[26] Rhee, K., Kwak, J., Kim, S. and Won, D. (2005). Challenge-response based RFID authentication protocol for distributed database environment. *Security in Pervasive*

Computing. LNCS. 3450: 70-84.

[27] Smart Card Alliance. (2008). Transit payment system security. *A Smart Card Alliance Transportation Council White Paper.* http://www.smartcardalliance.org/resources/lib/Transit_Payment_System_Security_WP.pdf

[28] Weis, S.A., Sarma, S.E., Rivest, R.L. and Engels, D.W. (2003). Security and privacy aspects of low-cost radio frequency identification systems. *Security in Pervasive Computing.* LNCS. 2802: 50-59.

[29] Yao, A.C. (1982). Theory and application of trapdoor functions. *23rd Annual Symposium on Foundations of Computer Science (FOCS 1982)*. pp. 80-91.