Authentication Issues in Near Field Communication and RFID

Submitted by

Muhammad Qasim Saeed

for the degree of Doctor of Philosophy

of the

Royal Holloway, University of London

2014

Declaration

I, Muhammad Qasim Saeed, hereby declare that this thesis and the work presented in it is entirely my own. Where I have consulted the work of others, this is always clearly stated.

Signed .... (Muhammad Qasim Saeed)
Date:

To Eve, who enjoys the most powerful status in information security.

Abstract

Near Field Communication is a short-range wireless technology based on RFID standard ISO 18092, ISO 14443 and ISO 15693. This means, it provides compatibility with the millions of contactless smartcards and RFID scanners that already exist worldwide. NFC is now available on the phones and this integration has resulted in a sharp rise in its utility. An NFC-enabled cell phone acts as an RFID reader to read compatible RFID tags (NFC tags), such as smart posters. The same cell phone can also be used as an NFC tag storing relevant data. In this case, a cell phone transforms into a digital wallet storing bank cards (money), vouchers, loyalties card etc., at a secure place called 'Secure Element'. Abuse of NFC technology is also on sharp rise because of large number of users and inadequate security standards. This thesis looks at security issues of NFC and RFID and provides mechanisms to improve the security features. NFC Forum (an association for developing NFC standards) released the signature specification in 2010 describing rules to digitally sign the NFC tag's contents. A part of the thesis covers the security related issues of the signature specification. Later in the thesis, a new specification for authenticating an NFC tag is proposed, including a framework of its implementation in a supply chain in order to detect counterfeit products. The thesis also includes a framework for NFC mobile wallet, where the Secure Element in the cell phone is only used for customer authentication and the banking credentials are stored in a cloud. At the end of the thesis, security analysis of an authentication protocol for low-cost RFID tags is described with multiple attacks resulting in full disclosure of secret keys.

Acknowledgement

Completion of this doctoral dissertation was possible with the support of several people. I would like to express my sincere gratitude to all of them. First of all, I am extremely grateful to my research supervisor, Dr. Colin Walter, Director of Distance Learning, Information Security Group (ISG), Royal Holloway University of London, for his valuable guidance, scholarly inputs and consistent encouragement I received throughout the research work. He always made himself available to clarify my doubts despite his busy schedules and I consider it as a great opportunity to do my doctoral programme under his guidance and to learn from his research expertise. Thank you Dr. Colin Walter, for all your help and support.

Some faculty members of the ISG have been very kind enough to extend their help at various phases of this research, whenever I approached them, and I do hereby acknowledge all of them. I thank Dr. Kostantinos Markantonakis, Smart Card Centre, ISG, Royal Holloway University of London for providing me the expert opinions related to my area of research. I thank Dr. Gerhard Hancke for his valuable suggestions and concise comments about my work. I also thank Dr. Carlos Cid and Dr. Chez Ciechanowicz for their kind support during my annual reviews.

I would like to thank Prof. Keith Martin, Director ISG, Royal Holloway University of London for providing me the financial support whenever I needed. It was all because of his support that I was able to attend academic conferences all over the world to share and gain knowledge.

The thesis would not have come to a successful completion, without the help I received from the staff of the ISG. I would like to thank Jon Hart, Tristan Findley and Lisa Nixon for their support related to IT services. I also thank Guillaume Subra, Emma Mosley and Jenny Lee for providing me the administrative support.

I am thankful to my colleague, Zeeshan Bilal, for many fruitful discussions, exchanging innovative ideas and scholarly interactions.

I would like to thank the Government of Pakistan for sponsoring my studies and my stay in London.

I owe a lot to my parents who encouraged and helped me at every stage of my personal and academic life, and longed to see this achievement come true.

I am very much indebted to my family, my wife Misbah Fatima, daughters Fakiha and Duhaa, who supported me in every possible way to see the completion of this work.

Above all, I owe it all to Almighty God for granting me the wisdom, health and strength to undertake this research task and enabling me to its completion.

Contents

List of Figures

List of Tables

List of Notation

K	Shared secret key
K_{pub}, K_{pr}	Public-Private key pair
K_{sign}, K_{ver}	Signing, verification key pair
R	Random number
$\oplus$	Bitwise XOR operation
$\vee$	Bitwise OR operation
$\wedge$	Bitwise AND operation
$A \rightarrow B : M$	A sends to B, message M
X	A bit string of arbitrary length L, represented as $x_{L-1} \cdots x_0$, where x_0 and x_{L-1} are the least significant and most significant bits respectively
$\text{HW}(X)$	Hamming weight of bit string X
$Rot(X, Y)$	Left rotation of argument X by $\text{HW}(Y)$ bits

List of Abbreviations

Acc_{ID}	Account ID of the customer
App_{ID}	Transaction approval message for customer account ID
AuC	Authentication Centre (subsystem of MNO)
BS	Base Station
CA	Certificate Authority
Cr_{req}	Credit Request Message
Cr_{app}	Credit Approval Message
DIDT_i	Tag's dynamic identity used in i^{th} authentication session
EPC	Electronic Product Code
IDR	Reader's static identity
IDT	Tag's static identity
IMSI	Internet Mobile Subscriber Identity
MNO	Mobile Network Operator
MSC	Mobile Switching Centre
MTD	MNO Transaction Department
NDEF	NFC Data Exchange Format
NFC	Near Field Communication
PI	Payment Information
PoS	Point of Sale
PUF	Physical Unclonable Function
RTD	Record Type Definition
SBAD	Shop Bank Account Details
TEM	Transaction Execution Message
TMSI	Temporary Mobile Subscriber Identity
TNF	Type Name Format

Executive Summary

The thesis focuses on the security related issues of Near Field Communication (NFC) and RFID. NFC is a short range wireless communication technology based on RFID standard 18092, ISO 14443 and ISO 15693 and compatible with the existing contactless smart cards. NFC tags are used in a variety of applications like product identification, smart posters, access control etc. The integration of NFC with cell phone technology has given a new dimension to the utility of NFC. Alongside a sharp increase in the number of its users, the abuse of this technology also poses serious challenges.

In this thesis, we focused on authentication issues in NFC and RFID. We subdivided authentication issues in two different categories.

1. **Tag Data Authentication.** This category deals with the authentication of the data stored on an NFC or RFID tag. In most of the scenarios, NFC and RFID tags are deployed in public places accessible to every person. For instance, a smart poster advertising an event is displayed in an open environment where users can touch their cell phones to the poster to access its contents. In such scenarios, alteration in the tag's contents poses a serious threat whereas copying the tag contents to another tag is beneficial. A part of the thesis deals with the mechanisms authenticating the contents of a tag and not the tag itself.

2. **Tag Authentication.** There are occasions where copying the contents of an NFC tag to another tag is undesirable. For instance, a signed NFC tag is attached to a medicine packet storing its chemical composition and expiry date. Any NFC reader can read its contents and verify it using the signature. However, a counterfeit medicine and counterfeit tag with the same data will also be authenticated. A mechanism to authenticate a tag can thwart such attacks.

Contributions

1. Our first contribution is related to the *Tag Data Authentication* category. The Near Field Communication Forum was formed in 2004 to advance the use of NFC

by developing specifications, ensuring interoperability among devices and services, and educating the market about NFC technology[1]. The NFC Forum released various technical specifications as a process of standardising NFC technology. One of the specification, the Signature specification, is aimed at providing data integrity and authenticity to contents of an NFC tag. However, soon after its release, various vulnerabilities were discovered. We fixed several vulnerabilities and suggested amendments in the signature specification. As the result, the NFC Forum released a revised version of the signature specification in 2013. This work was published in the 6^{th} International Conference for Internet Technology and Secured Transactions, 2011 [62] and in the International Journal for Information Security Research (IJISR), 2012 [63]. The details are covered in Chapter 3.

2. Regarding the *Tag Authentication* category, the NFC Forum does not provide any specification to authenticate an NFC tag. The lack of such a mechanism opens the door to many security threats, particularly to counterfeit products when NFC technology is used in product identification. We addressed this weakness by proposing a specification, based on the data structure specified by the NFC Forum, that authenticates a tag along with its data. We provided justifications and role of the each field of the proposed specification. The main advantage of our proposed specification is its compatibility with the existing NFC devices. This work is published in the International Conference for Internet Technology and Secured Transactions 2012 [64]. The details are covered in Chapter 4.

3. After proposing the specification, we designed a framework to demonstrate its use in detecting counterfeits products in a supply chain. We proposed that if the products are equipped with NFC tags, the counterfeit products can be identified by authenticating the attached NFC tags. This is a customer-level framework so a customer can identify a counterfeit product by reading the NFC tag attached to the product with his cell phone. This work is published in the International conference on Privacy, Security and Trust (PST 2013) [61]. The details are covered in Chapter 5.

4. Tag authentication is of crucial importance when a tag is registered against a specific user or product. In such cases, the tag authentication leads to a user or product authentication. For instance, the access control tags serve as a tool for user authentication. We proposed two payment solutions, based on NFC, where a customer is identified and authenticated from his mobile device. After

[1]http://nfc-forum.org/

a successful customer authentication, the money is transferred from customer's to shopkeeper's account. This work is published in the International Journal of Advanced Computer Science and Applications (IJACSA), 2013 [56] and in the Eighth International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies, 2014 [60]. The details are covered in Chapter 6.

5. I also participated in cryptanalysis of an ultra-lightweight authentication protocol used for authentication of RFID tags. My contribution in this work is the Full Disclosure Active (FDA) Attack on the authentication protocol. This attack reveals the secret key stored in the RFID tag. This work is published in the Journal of Applied Mathematics and Information Sciences [17]. The details are covered in Chapter 7.

List of Publications

[1] Muhammad Qasim Saeed and Colin D. Walter. A Record Composition / Decomposition Attack on the NDEF Signature Record Type Defnition. In *The 6th International Conference for Internet Technology and Secured Transactions (ICITST-2011)*, pages 283-287, IEEE, Dec 2011.

[2] Muhammad Qasim Saeed and Colin D. Walter. An Attack on Signed NFC Records and Some Necessary Revisions of NFC Specifications. In *The International Journal for Information Security Research (IJISR)*, pages 326-334, March 2012.

[3] Muhammad Qasim Saeed and Colin D.Walter. Off-line NFC Tag Authentication. In *The 7th International Conference for Internet Technology and Secured Transactions (ICITST-2012)*, pages 730-735, IEEE, December 2012.

[4] Muhammad Qasim Saeed, Zeeshan Bilal, and Colin D. Walter. An NFC based consumer-level counterfeit detection framework . In *The 11th International Conference on Privacy, Security and Trust (PST 2013)*, pages 135-142, IEEE, July 2013.

[5] P. Pourghomi, M.Qasim Saeed, and G. Ghinea. A Proposed NFC Payment Application. In *The International Journal of Advanced Computer Science and Applications (IJACSA)*, volume 4, pages 173-181, SAI, 2013.

[6] M.Qasim Saeed, Colin D. Walter, P. Pourghomi, and G. Ghinea. Mobile Transactions over NFC and GSM. In *The Eighth International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies, UBICOMM, Italy* (To appear), IEEE, 2014.

[7] Zeeshan Bilal, Keith Martin, and Qasim Saeed. Multiple Attacks on Authentication Protocols for Low-Cost RFID Tags. In *Journal of Applied Mathematics and Information Sciences* (To appear). Natural Sciences, 2014.

Part I

Background

Chapter 1

Background to RFID and NFC Technologies

This chapter provides an introduction to NFC and RFID systems. It also provides an overview of how the NFC technology over cell phones can be used in everyday life.

1.1 Introduction

Radio Frequency Identification (RFID) is a wireless technology that enables identification of tags over a radio link. In recent years, RFID technology has moved from obscurity into mainstream applications. RFID is used for identification in a wireless manner and, unlike earlier bar-code technology, it does not require a line of sight. This technology consists of an RFID tag attached to an object, and an RFID reader that reads the RFID tags to identify it. Moreover, RFID tag can store additional data (such as manufacturer details, expiry date etc.,) than the barcodes. These characteristics make RFID a suitable technology for identification of products in supply chain, identification of human for access control or in any other framework where identification is of crucial importance.

RFID tags are of various types, but at the highest level, the tags are divided into two main classes [69]:

- Active Tags
- Passive Tags

Active tags require a power source. They are either connected to a powered infrastructure or equipped with a battery. This not only increases their cost and size but also reduces their usability.

In contrast, Passive tags do not require a dedicated power source to operate. These tags receive power from the reader in order to perform computations and transmit data. Characteristics like low-cost, maintenance-free, small size with indefinite operational life make them more practicable to be used for identification purpose.

Our focus is only on passive tags as the thesis covers authentication issues of such tags. Since these tags lack an inbuilt battery, they utilize power received from the reader.

1.2 Power for Passive RFID Tags

There are two main approaches exist to transfer power from a reader to a passive tag [69]:

1. Electromagnetic (EM) wave capture
2. Magnetic induction

Both approaches can transfer enough power to a remote tag to compute and transmit back the response. Depending upon the power up approach, the tag can be either a *Far-field* or a *Near-field* tag.

1.2.1 Far-Field RFID Tags

The *Far-field* RFID tags receive power by capturing EM waves of the reader as shown in Figure 1.1 [69]. A simple electronic circuit in the tag is used to accumulate the energy in order to power up its electronics. The energy a tag captures is a small fraction of the transmitted energy from the reader. The energy is so acute that the tag is unable to *transmit* back the response with enough strength to reach it to the reader; rather it *back-scatters* the reader's transmission in a pattern that stores some of its information like Tag ID etc. Generally, far-field tags operate in the ultra-high frequency (UHF) band (such as 2.45 GHz). A typical far-field reader can successfully interrogate tags 3m away, and some RFID companies claim their products have read ranges of up to 6m [69].

The limited amount of power also restricts the computational capabilities of far-field tags. These tags can only perform simple operations like XOR, bit rotations etc., but are unable to perform heavy cryptographic computations. This results in many security related issues in such tags. A special branch of cryptography, *light-weight cryptography*, deals with the security issues of these tags but unlike conventional cryptography, this

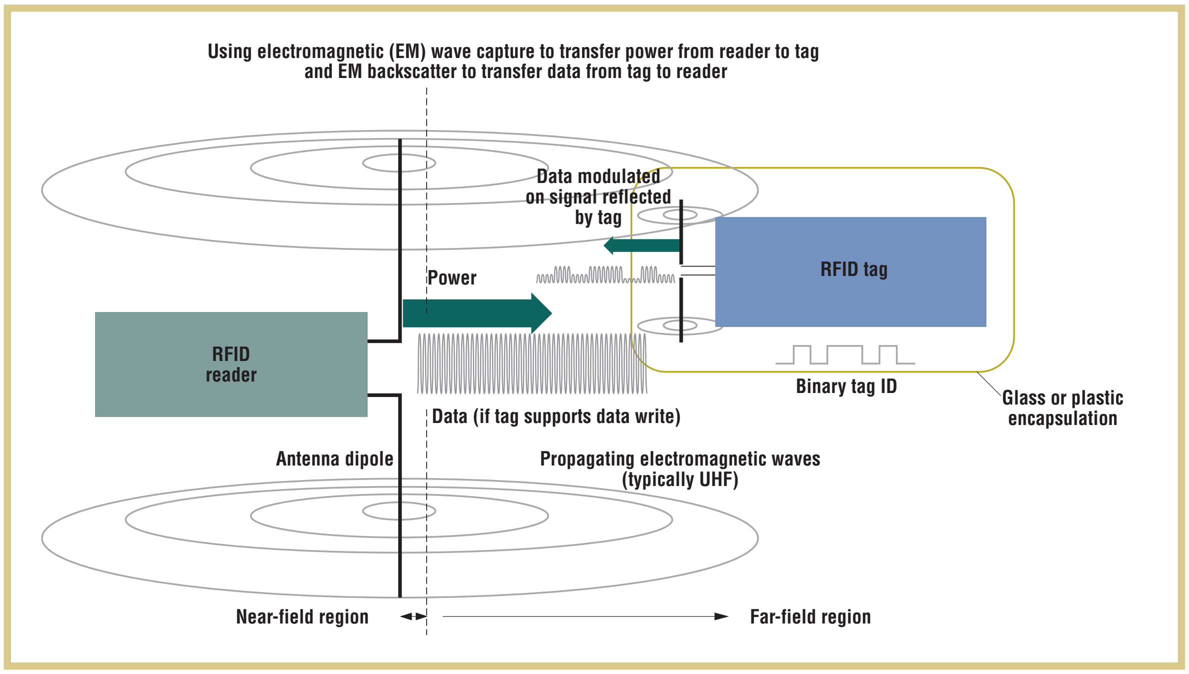
Using electromagnetic (EM) wave capture to transfer power from reader to tag
and EM backscatter to transfer data from tag to reader
Data modulated on signal reflected by tag
RFID tag
Power
RFID reader
Binary tag ID
Glass or plastic encapsulation
Data (if tag supports data write)
Antenna dipole
Propagating electromagnetic waves (typically UHF)
Near-field region
Far-field region

Figure 1.1: Far Field RFID Technology

branch yet needs to mature. Various security algorithms and protocols for light-weight cryptography are under development [48].

We also contributed in development of authentication protocols for these tags. This contribution is in the form of a security analysis of earlier proposed authentication protocols. The details are described in Chapter 7.

A well-known example of far-field RFID tag is Electronic Product Code (EPC) tags used for for product identification in supply chains [65]. We will discuss these tags in Chapter 5.

1.2.2 Near-Field RFID Tags

In *Near-field* RFID tags, the power is transferred from the reader through Faraday's principle of magnetic induction. A reader generates alternating magnetic field in its locality by passing a large alternating current through its coil. A tag placed inside the alternating magnetic field will develop a voltage across the tag's coil as shown in Figure 1.2 [69]. This voltage is used to power the tag chip.

Near-field coupling is the most straightforward approach for implementing a passive RFID system. This is why it was the first approach taken and has resulted in many subsequent standards, such as ISO 18092 and 14443, and a variety of proprietary solutions.

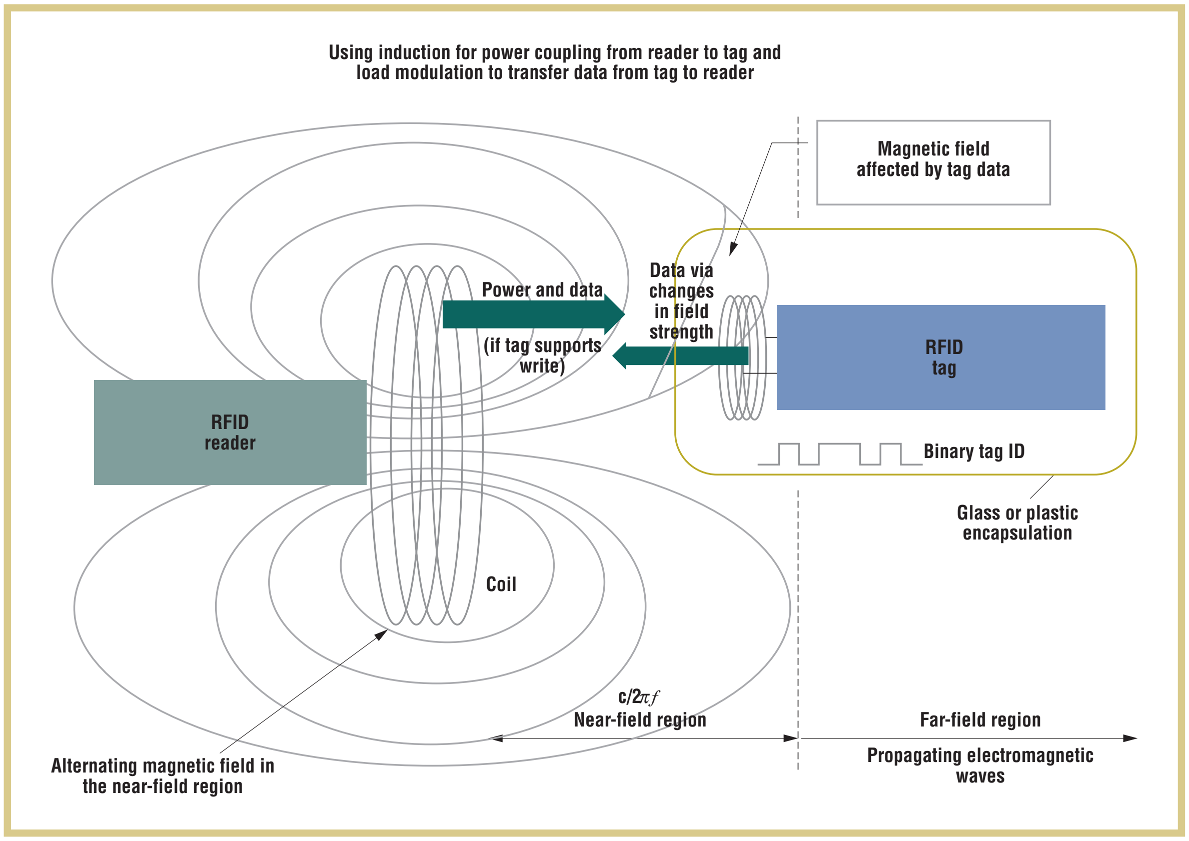
Using induction for power coupling from reader to tag and load modulation to transfer data from tag to reader
Magnetic field affected by tag data
Data via changes in field strength
Power and data
(if tag supports write)
RFID tag
RFID reader
Binary tag ID
Glass or plastic encapsulation
Coil
c/2πf
Near-field region
Far-field region
Propagating electromagnetic waves
Alternating magnetic field in the near-field region

Figure 1.2: Near Field RFID Technology

1.3 Near Field Communication

The Near-Field RFID forms the basis of Near Field Communication (NFC). NFC is often seen as an extension to the near-field RFID. Like an ordinary reader, the NFC devices can read RFID tags based on specific standards; but unlike RFID technology, two NFC devices can communicate to each other in a peer-to-peer mode, or one NFC device can act as an RFID tag and other as a reader. NFC operates at 13.56MHz frequency band with an operational distance of less than 10 cm. Possible supported data transfer rates are 106, 212 and 424 kbps. NFC's bidirectional communication ability is ideal for establishing connections with other technologies by the simplicity of a touch[1].

NFC was developed by Philips and Sony in 2002. Many existing standards from RFID were adopted in the NFC. The NFC base standard for the physical layer, NFCIP-1 (Near Field Communication Interface and Protocol 1), is standardized in ISO 18092 [36] and ECMA 340 [26]. This standard specifies the basic characteristics, such as transfer speeds, coding, modulation schemes, frame architecture, and RF interface. It also provides initialization schemes and conditions that are required to prevent collisions during initialization. The RF layer used in the NFCIP-1 is directly inherited from the older ISO 14443 (Proximity Contactless Cards), more specifically the Type A protocol defined in that standard, and on Japanese JIS 6319-4 (on which Sony FeliCa is based). Consequently, NFC devices (reader/writer mode) are compatible with ISO 14443 smart cards.

The second major standard is NFCIP-2 (ISO 21481 [37] or ECMA 352 [27]), which defines the selection mechanism between different contactless technologies that operate on the same frequency 13.56MHz. It is intended to be used by mobile devices that support communication according to ISO 18092, ISO 14443, but they must also be compatible with other contactless standards like ISO 15693.

NFC devices have to provide ISO 14443 (proximity cards, e.g. Philips Mifare), ISO 15693 (vicinity cards) and Sonys FeliCa contactless smart card system in order to be compatible with the main international standards for smartcard interoperability. Hence, as a combination of smartcard and contactless interconnection technologies, NFC is compatible with today's field proven RFID-technology. That means, it is providing compatibility with the millions of contactless smartcards and scanners that already exist worldwide.

In 2004 Nokia, Philips and Sony established the Near Field Communication (NFC) Forum[2] to advance the use of NFC technology by developing specifications, ensuring interoperability among devices and services, and educating the market about NFC technology. Some of the standards developed by the NFC Forum, of concern to our area of research, are described in Chapter 2.

1.3.1 NFC on Cell Phones

The integration of NFC with cellular technology resulted in a sharp rise in its utility. The leading mobile phone manufacturers like Samsung, Nokia, HTC, Sony are integrating NFC in most of their devices. Notably absent among them is Apple with its iPhone. A frequently updated list of the cell phones equipped with NFC is available at

[1]http://nfc-forum.org/
[2]http://nfc-forum.org/

Figure 1.3: "N-Mark" logo used to identify NFC tags and devices

the NFC World website[3] and is also attached as Appendix A to the thesis.

There can be a variety of applications of NFC over cell phones. Imagine a busy lady who leaves her house in the morning; she secures her apartment door by touching her phone to the door knob. On her way to the train station, she purchases a coffee by touching her phone to the payment terminal at the coffee shop. At the station, she touches her phone to the turnstile to debit her fare from her transit account. After she is seated on the train, she sees a poster for an upcoming concert she wants to attend. She touches her phone to the poster to transfer the event details to her calendar and purchase tickets. Arriving at work, she touches her phone to the door to enter the building. On her coffee break, she buys a snack from the vending machine by touching her phone to the payment panel. While meeting with a client, she exchanges contact details by touching her phone to the client's. After work, she meets a friend and shares the details of the upcoming concert by touching her phone to her friend's. All of these things are possible because of the integration of NFC with cellular technology [34].

An NFC equipped cell phone can act as an RFID reader to read from or write to a tag. Additionally, two cells phones can share data in a peer-to-peer mode or a cell phone can act as a tag (card emulation mode) which is read by any compatible reader.

NFC architecture on a mobile phone consists of the following components [47]:

1. **NFC Antenna.** This is responsible for the physical connection of the cell phone to another RF reader, NFC device or RF tag.

2. **NFC Controller.** This converts analog to digital or *vice versa* for transmission and receiving through the NFC antenna. Additionally, it controls all processes related to the NFC in a cell phone.

3. **Secure Element.** In card emulation mode, a Secure Element (SE) is used to store sensitive data. The SE is a combination of hardware, software, interfaces and protocols that enable secure data storage and application execution. The

[3]http://www.nfcworld.com/nfc-phones-list/

cell phone equipped with SE can be used for services requiring a high level of security such as payment application, ticketing, etc. The SE can be in a form of removable (SIM, external memory card) or non-removable (embedded in hardware) component within the cell phone, or it can be in a cloud providing all the necessary services.

4. **Host Controller.** The Host Controller interacts with the NFC Controller and in some cases, with the Secure Element as well (for instance, to top-up credit in the Secure Element over the air).

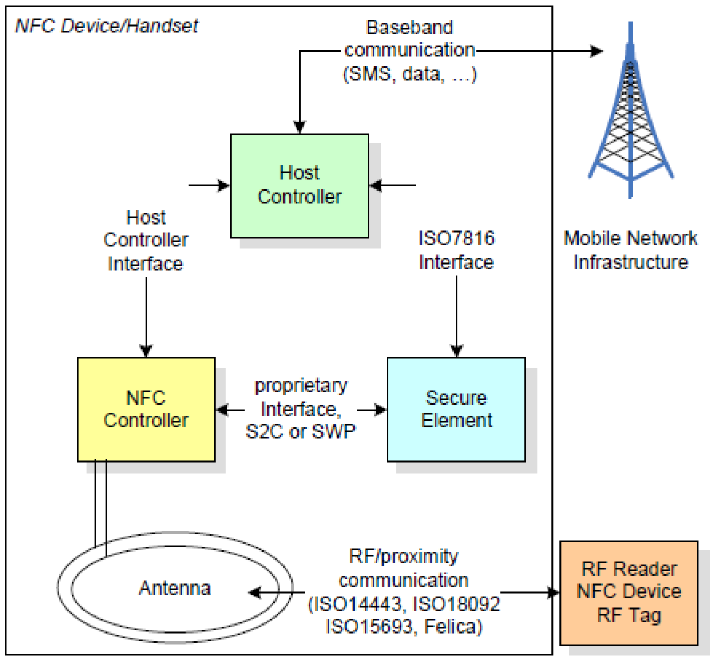
NFC Device/Handset
Baseband communication (SMS, data, ...)
Host Controller
Host Controller Interface
ISO7816 Interface
Mobile Network Infrastructure
NFC Controller
proprietary Interface, S2C or SWP
Secure Element
Antenna
RF/proximity communication (ISO14443, ISO18092 ISO15693, Felica)
RF Reader NFC Device RF Tag

Figure 1.4: Architecture of NFC integrated in a mobile device

A basic NFC architecture of NFC in a mobile device is illustrated in Figure 1.4 [47].

Managing the SE among various NFC stakeholders (Banks, cell phone manufacturers, Mobile network operators, merchants etc.) is an arduous job because of lack of trust and business limitations. This area of NFC is yet to mature as it lacks a standard infrastructure to be followed by all stakeholders.

Although the primary uses of NFC are likely to be commercial, users can also purchase their own tags and use them to automate certain tasks. For example, users could mount a tag on their nightstand that sets their phone's alarm function. Programming a tag with your home wireless network details would allow guests to connect by touching their phones to the tag. Placing a tag in your car or on your keychain can switch on the Bluetooth to pair your phone with your car's stereo system. Tags can also be set in a 'toggle' mode that will either set or revert to a group of settings on the cell phone each time the tag is scanned.

1.3.2 Gartner Hype Cycle for NFC

Gartner Hype Cycles, a branded graphical tool developed and used by IT research and advisory firm *Gartner*, provides a graphic representation of the maturity and adoption of technologies and applications, and how they are potentially relevant to solving real business problems and exploiting new opportunities[4].

Each Hype Cycle drills down into the five key phases of a technology's life cycle.

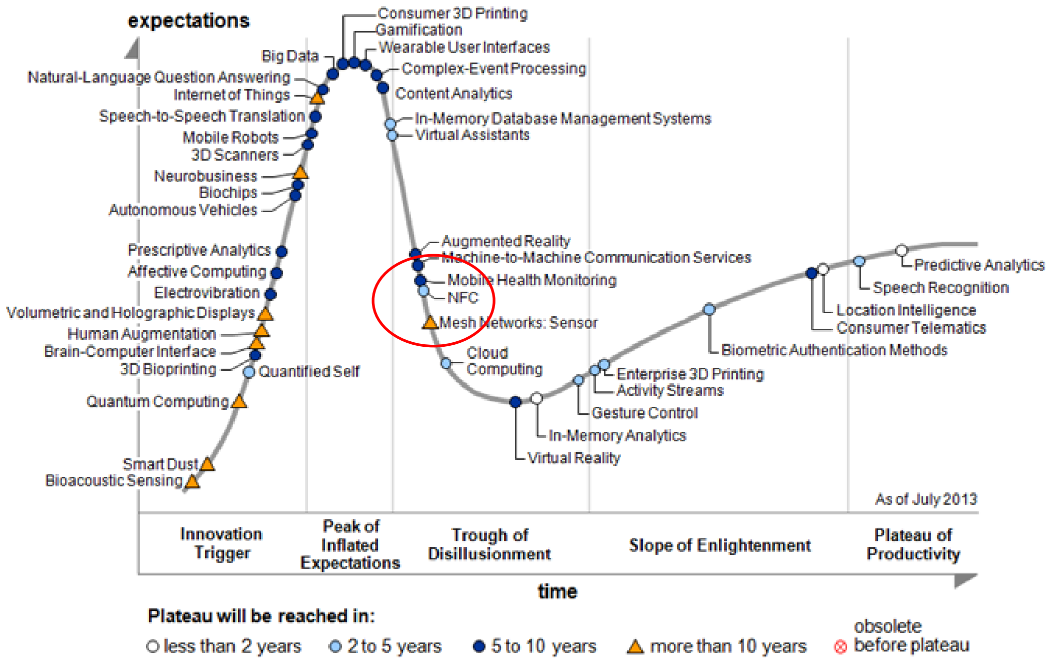

Figure 1.5: Gartner Hype Cycle 2013

1. **Technology Trigger:** A potential technology breakthrough kicks things off. Early proof-of-concept stories and media interest trigger significant publicity. Often no usable products exist and commercial viability is unproven.

2. **Peak of Inflated Expectations:** Early publicity produces a number of success stories often accompanied by scores of failures. Some companies take action; many do not.

[4]http://www.gartner.com/technology/research/hype-cycles/

3. **Trough of Disillusionment:** Interest wanes as experiments and implementations fail to deliver. Producers of the technology shake out or fail. Investments continue only if the surviving providers improve their products to the satisfaction of early adopters.

4. **Slope of Enlightenment:** More instances of how the technology can benefit the enterprise start to crystallize and become more widely understood. Second- and third-generation products appear from technology providers. More enterprises fund pilots; conservative companies remain cautious.

5. **Plateau of Productivity:** Mainstream adoption starts to take off. Criteria for assessing provider viability are more clearly defined. The technology's broad market applicability and relevance are clearly paying off.

The NFC is currently (2013) in the Trough of Disillusionment phase and needs 2 to 5 years to reach the Plateau of Productivity as shown in Figure 1.5.

This implies that after a couple of years, the NFC would become a widely implemented main-stream technology provided that the NFC stake holders improve their products and standards to pull the NFC out of the Trough of Disillusionment. Well-defined standards and well-understood applications are the key factors for the success of the NFC technology. Since NFC will be mostly used for m-commerce, the security is the back-bone of the NFC. However, there are still many security related issues that need to be addressed.

1.4 Security Concerns

Being a wireless technology, the NFC suffers not only from conventional security threats, such as eavesdropping, data insertion, data modification, man-in-the-middle attack, DoS attack etc. [33, 47]; but also faces a new threat from another dimension: NFC-enabled cell phones. Conventional attacks in any RFID system generally require an observable attacking platform, such as a laptop attached to a specific hardware, or custom-built card emulators and off-the-shelf readers. This makes the attacker suspicious in public thus making the attacks arduous. But things are not the same in NFC technology. Here, the attacker just needs an NFC-enabled cell phone as an attacking platform to launch an attack. Unlike a laptop or a card-emulator, holding a cell phone near to any Point of Sale terminal would be accepted by merchants and arouse less suspicion in public. Lishoy Francis *et al.* demonstrated that a cell phone can be used as a pick-pocketing tool [30]. They performed two attacks, token cloning and contactless skimming, using an NFC-enabled cell phone. Same researchers also performed a peer-to-peer relay attack using NFC-enabled cell phones [31]. They demonstrated that a contactless card in a pocket can be read without any intimation to the person by holding an NFC-enabled cell phone near to pocket of the person. Similarly, e-Passports can also be read with the cell phone without any consent of its owner. The data is not only skimmed from the contactless card or from an e-Passport to the NFC-enabled cell phone, but also relayed to any other machine in real time.

We, at this stage, contributed in the development of security related standards of the NFC technology to help it reach at the Plateau of Productivity. For instance, we highlighted some vulnerabilities in the Signature Specification, a guideline used to digitally sign the contents of an NFC tag. We also suggested some improvements in the Signature Specification and submitted our suggestions to the NFC Forum, an organization responsible for the standardisation of the NFC technology. This resulted in a revised signature specification (more details in Chapter 2 and 3).

We also realized that the NFC Forum does not provide any specification to authenticate an NFC tag. We proposed a new specification for this purpose and provided a framework to implement the proposed specification in a supply chain to detect counterfeit products (Refer Chapters 4 and 5).

Keeping in view the likely use of NFC to be commercial, we also proposed a secure mobile payment solution described in Chapter 6.

Authentication of lightweight RFID tags is a challenging task as the these tags are unable to perform standard cryptographic functions. We analyzed a mutual authentication protocol in Chapter 7 and showed that the protocol is vulnerable to key recovery attacks.

Chapter 2

The NFC Forum

This chapter provides information about the NFC Forum and two of its most important specifications: The NFC Data Exchange Format (NDEF) and the Signature Record Type Definition.

2.1 Introduction

The Near Field Communication Forum was formed in 2004 to advance the use of Near Field Communication technology by developing specifications, ensuring interoperability among devices and services, and educating the market about NFC technology[1]. Manufacturers, applications developers, financial services institutions, and more all work together to promote the use of NFC technology in consumer electronics, mobile devices, and PCs. The NFC Forum promotes sharing, pairing, and transactions between NFC devices or tags. The goals of the NFC Forum are:

- Develop standards-based Near Field Communication specifications that define a modular architecture and interoperability parameters for NFC devices and protocols.
- Encourage the development of products using NFC Forum specifications.
- Work to ensure that products claiming NFC capabilities comply with NFC Forum specifications.
- Educate consumers and enterprises globally about NFC.

In June 2006, the Forum formally outlined the architecture for NFC technology. The Forum has released 18 specifications to date[2]. The specifications provide a road

[1]http://nfc-forum.org/
[2]June, 2014

map that enables all interested parties to create powerful new consumer-driven and developer-driven products. These specifications can be viewed at the NFC Forum website.

2.2 NFC Forum Tags

NFC Tags are integrated circuits storing data that can be read by NFC-enabled devices. In order to maintain the interoperability of NFC devices and tags, the NFC Forum has specified four different types of tag [51]:

- **Types 1 & 2:** These tags are read and re-write capable. Users can configure the tag to become read-only. Memory availability is 48 or 96 bytes, expandable to 2 KB.
- **Type 3:** These tags are based on the Japanese Industrial Standard (JIS) X 6319-4 known as FeliCa. They are pre-configured at manufacture to be either read and rewritable, or read-only. Their memory has a theoretical limit of 1 MB per service.
- **Type 4:** These tags are pre-configured at manufacture to be either read and re-writable, or read-only. There is up to 32 KBytes of memory available per service. The Application Protocol Data Unit (APDU) based on ISO/IEC 7816-4 is used for communication between tag and reader.

2.3 NFC Forum Specifications

Considering the security aspect of NFC technology, we will discuss following two specifications, out of 16, in details in this chapter.

1. **NFC Data Exchange Format (NDEF) Technical Specification**. It specifies a common data format for NFC Forum-compliant devices and NFC Forum-compliant tags.
2. **NFC Signature Record Type Definition (RTD) Technical Specification**. The signature record provides the integrity and authenticity to an NDEF message by digitally signing its contents. The signature record specifies the format used when signing single or multiple NDEF records. It defines the structure of the signature record, provides a list of suitable signature and hashing algorithms, and certificate types that can be used to create the signature.

2.3.1 NFC Data Exchange Format

The NFC Data Exchange Format (NDEF) defines a common format and rules to exchange information in the NFC environment. The NFC Forum released NDEF Version 1.0 in July, 2006. It defines the NDEF data structure format as well as rules to construct a valid NDEF message as an ordered and unbroken collection of NDEF records. NDEF is a lightweight, binary message format that can be used to encapsulate one or more application-defined payloads of arbitrary type and size into a single message construct. Each payload is described by a type, its length, and an optional identifier. A record is the unit for carrying the payload within an NDEF message. An NDEF message contains one or more NDEF records as shown in Figure 2.1.

NDEF Message						
R_1, *MB*=1	…	R_r	…	R_s	…	R_t, *ME*=1

Figure 2.1: An NDEF message with a set of records R_1 to R_t

The structure of an NDEF record is shown in Figure 2.2. Each row represents a single byte.

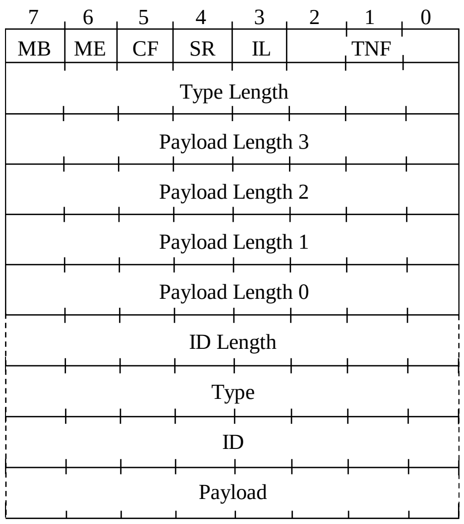

Figure 2.2: NDEF Record Layout

NDEF Header

NDEF Header is the first byte of an NDEF record. Message Begin ***(MB)*** and Message End ***(ME)*** flags mark the first and the last record of an NDEF message respectively. The Chunk Flag ***(CF)*** bit specifies that the payload of that record is continued in the next record. Short Record ***(SR)*** is a 1-bit flag which, if set, indicates that the size of the *Payload-Length* field is one byte. In this case, the payload size is restricted to between 0 and 255 bytes. Otherwise, the *Payload-Length* field consists of 4 bytes (as shown in Figure 2.2) and the Payload size ranges from 0 to $2^{32}-1$ bytes. The flag ***IL*** determines whether or not the optional *ID* field and corresponding *ID-Length* field are present.

The Type Name Format ***(TNF)*** is a 3-bit field indicating the structure of the *Type* field. Its value ranges between 0 and 7 as shown in Table 2.1.

TNF	Description
0	The record is empty and there is no payload or type associated with this record. The corresponding length fields are set to zero. This TNF value can be used whenever an empty record is needed.
1	Indicates that the *Type* field contains a value that follows the RTD type name format defined in the NFC Forum RTD specification, such as Smart poster RTD, Signature RTD, URL RTD etc.
2	Type is a MIME media type identifier (RFC 2406).
3	Type is an absolute URI (RFC 3986).
4	Type is an NFC Forum external type.
5	Type is of unknown format. It is used when the type of the payload is unknown. When used, the *Type-Length* field must be zero and thus the *Type* field is omitted. In this case, the payload is stored but not processed.
6	The record continues the payload of the preceding chunked record. When used, the *Type-Length* field must be zero and thus the *Type* field is omitted.
7	Reserved for future use.

Table 2.1: Type Name Format *(TNF)* Description

Type-Length and ***ID-Length*** are unsigned 8-bit integers that specify the length in octets of the *Type* field and *ID* field respectively.

Payload-Length field is an unsigned integer that specifies the length in octets of the Payload field. The size of the *Payload-Length* field is 4 bytes when the *SR* flag is clear, and otherwise the size is 1 byte. By providing the payload length within the first few bytes of a record, efficient record boundary detection is possible.

The ***Type*** field describes the type of the payload. NDEF supports URIs, MIME media type constructs, and NFC-specific type formats as the type identifiers. By indicating the type of a payload, it is possible to dispatch the payload to the appropriate user application.

The ***ID*** field is an optional identifier in the form of an absolute or relative URI.

The use of an identifier enables payloads that support URI linking technologies to cross-reference other payloads.

Record Chunks

A record chunk carries a chunk of a payload. It can be used to partition dynamically generated contents or very large entities into multiple subsequent record chunks within an NDEF message. Every chunk payload in encoded as an *initial* record chunk followed by zero or more *middle* record chunks and finally terminated by a *terminating* record chunk [3].

The initial record chunk has its *CF* flag set. The *Type* field and the *ID* field (if present) indicate the type and ID of the entire payload respectively. The *payload-length* field indicates the size of payload of the initial record only.

The *middle* and *terminating* record chunks do not have *Type* and *ID* fields as these are already indicated in the *initial* chunk. Their *TNF* field value is 6, indicating that the *Type* and *ID* are the same as for the *initial* record chunk. Their *Type-length* and *ID-length* fields are zero. The *CF* is set for middle chunks and is clear for the terminating chunk.

A chunked payload is entirely encapsulated within a single NDEF message. As a consequence, neither an initial nor a middle record chunk can have the *ME* flag set.

2.3.2 Signature Record Type Definition

With the increasing number of available applications of NFC technology, threats of its abuse also emerged in parallel. Lack of any mechanism to provide data integrity to NDEF messages paved the way to exploit NDEF messages. A soft target for such attacks is the NFC tag, a chip storing data that can be read in a wireless manner by an NFC reader. NFC tags are generally deployed in open environment, like smart posters, where they are subject to physical attacks like data alteration or tag replacement. Smart posters contain information such as Title, SMS, and a URL or electronic business card. The user can access this information by simply touching the cell phone on such tags. Apart from displaying the information to the user, the smart poster can also trigger an action such as opening a specific website, calling the telephone number stored in the poster etc [4]. An attacker may replace the URL address or the telephone number in a smart poster with malicious content. Eventually, the reader is diverted to malicious contents as the former has no mean to ascertain the integrity of the message.

To address such issues, the NFC Forum developed the Signature Record Type Definition Version 1.0 in 2010 [7]. The main objective of the signature RTD is to digitally sign the data of an NDEF message thus providing integrity and authenticity. The signature is applied to a selection of the data of an NDEF message, and not to the whole NDEF message. Soon after its release, various attacks were launched because of the '*partially signed*' NDEF messages. We provided countermeasures to those attacks by adding more, but not all, fields to the signature (explained in Chapter 3) and communicated our recommendations to the NFC Forum. Consequently, the signature specification in under revision and the candidate specification of its upgraded version, V2.0, has been released in April 2013 for feedback and comments [9]. The major difference in the version 2.0 is that the signature is applied to all fields as compared to a selection of fields in the previous version. Although inclusion of all fields makes the signature specification more secure, it results in implementations issues. These issues are discussed in Chapter 3, §3.5. Table 2.2 compares the fields that are included in signature in the V1.0 and the V2.0. The version 2.0 is yet to be finalized.

Since the V1.0 is still valid and our work is related to the same version, we will refer to the same version throughout our thesis unless mentioned otherwise.

Structure of the Signature Record

The Signature Record is structured according to the format of an NDEF record shown in Figure 2.2. The *MB, ME, CF, SR* and *IL* flags of the Signature Record are set/unset according to the requirements. The 3-bit TNF value is `001` indicating that the record follows the RTD type name format defined in the NFC Forum RTD specification as mentioned in Table 2.1. The value of the Type-Length field is `0x3` indicating the Type field consists of three bytes (`0x53, 0x69, 0x67`) corresponding to the word "Sig".

Payload of the signature record stores the signature and corresponding certificates. The contents of the payload of a signature record consist of three parts: *Version, Signature* and *Certificate chain* as shown in Figure 2.3. The *Version* is a single byte field indicating the version of the specification to which a signature is compliant. The *Signature* field contains either the actual signature or a URI reference to a signature. The signature RTD supports RSA, DSA and ECDSA. The *Certificate Chain* contains the certificate format, the total number of certificates, the list of certificates and an optional URI reference. The *Certificate Store* sub-field of the Certificate Chain does not contain the top-most certificate in the certificate hierarchy (e.g., the root Certificate Authority (CA) certificate in an X.509 certificate chain). Since certificate validation requires that the top-most certificate in the certificate hierarchy be distributed independently, this specification omits the top-most certificate, under the assumption that the NFC Forum

Device must already possess it in order to validate the chain.

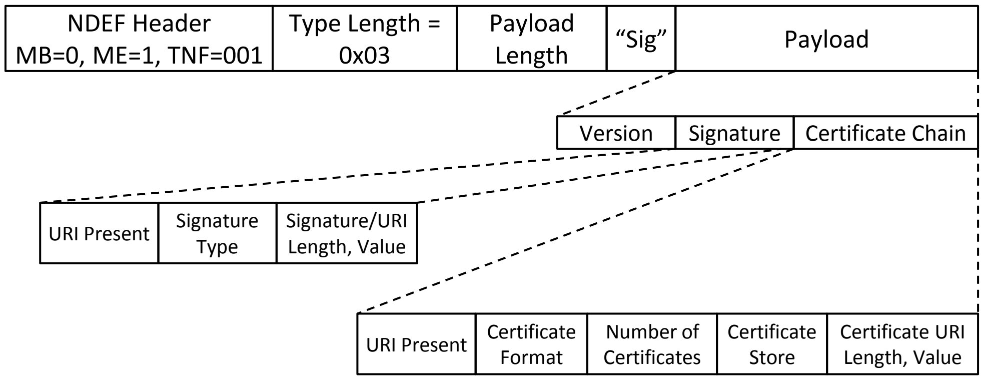

Figure 2.3: Structure of NDEF Signature record

Use of the Signature Record

The Signature record is applied to all preceding records, starting either from the first record of the NDEF message or from the first record following the preceding Signature Record as shown in Figure 2.4. In that figure, Signature Record 1 signs Records 1 and 2. It also marks the start of the signature of Record 3. Signature Record 2 signs Record 3 only whereas Record 4 is not signed. The signature is applied to the *Type*, *ID* (if present) and *Payload* of these records. Table 2.2 describes the fields of a record that are included or excluded from the signature. The Signature Record itself is not signed.

Field Name	V 1.0	V 2.0
Message Begin *(MB)*	Not signed	Signed
Message End *(ME)*	Not signed	Signed
Chunk Flag *(CF)*	Not signed	Signed
Short Record *(SR)* Flag	Not signed	Signed
ID-Length *(IL)* Present Flag	Not signed	Signed
Type Name Format *(TNF)*	Not signed	Signed
Type-Length	Not signed	Signed
Payload-Length	Not signed	Signed
ID-Length	Not signed	Signed
Type	Signed	Signed
ID	Signed	Signed
Payload	Signed	Signed

Table 2.2: Comparison of Signature Record V1.0 and 2.0

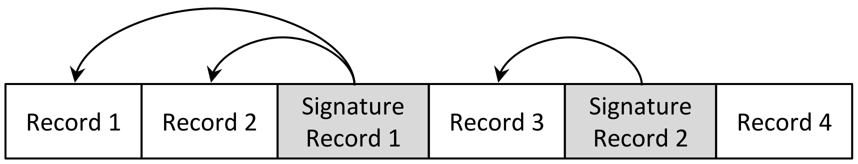

Figure 2.4: An NDEF message consisting of multiple records.

MB flag is intentionally unsigned so that a group of signed NDEF records may be moved to any position within an NDEF message [59]. This enables a variety of messages to be constructed for different target viewers around an important core content which is signed. However, it is unnecessary to sign the *ME* flag as the end of the section of the message which needs to have its integrity secured is marked by the signature record, so all the records preceding the signature record will have $ME = 0$.

A principle requirement of the signature definition is to be able to partition an NDEF record into multiple record chunks or *vice versa* as shown in Figure 2.5 without affecting the validity of the signature. This means, in particular, that only the *Payload* is to be included in the signature for records after the first chunk. Since the signature record is applied to *Type*, *ID* and *Payload* of a record, is provides a way to fulfill this requirement as the non-initial chunks do not have *Type* and *ID* fields as described in § 2.3.1.

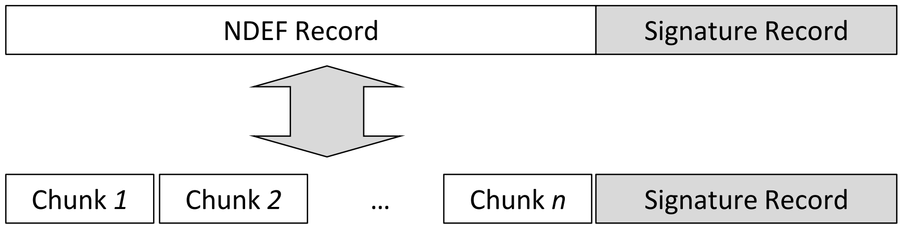

Figure 2.5: Record chunks with digital signature

Thus, the signature record does not provide integrity protection to all fields of a record, but to a selection of fields. The *'partially signed'* NDEF records were later exploited and multiple attacks were launched on the signature record. The details of such attacks will be discussed in the next chapter.

2.4 Summary

This chapter provides details about NFC Forum and two of its most important specifications; NFC Data Exchange Format (NDEF) specification and Signature Record specification. NDEF defines a common format to exchange information between NFC Forum devices and NFC Forum tags. The signature record provides integrity and authenticity to the data exchanged by NDEF messages. It achieves this by digitally signing NDEF messages. It also provides rules to sign and store the signature in the form of an NDEF record.

The signature is computed over *Type*, *ID* (if present) and *Payload* as described in §2.3.2. The *partially signed* NDEF record can lead to various attacks. The next chapter describes such attacks in detail followed by our proposed countermeasures.

Part II

Some Contributions

Chapter 3

Attacks on NDEF Specification

This chapter contains some attacks on the NDEF signature specification and some countermeasures to such attacks.

3.1 Introduction

The Signature specification released by the NFC Forum is aimed at providing data integrity and authenticity to NDEF messages. It achieves this goal by adding a digital signature and corresponding certificates to the NDEF message. The signature specification, however, does not provide data integrity to all fields of an NDEF message as discussed in the previous chapter. The *'partially signed'* messages are vulnerable to multiple attacks such as Record Composition Attack and Record Decomposition Attack. These attacks were highlighted by M. Roland in 2011, soon after the release of the signature specification [50].

The Record Composition Attack is aimed at composing different records in such a way that the digital signature remains valid. There are two scenarios described by M. Roland to accomplish this attack.

In the first scenario, two different NDEF messages are selected in which every record has its own signature. A malicious NDEF message can be created by selecting only a few of the records along with their signatures from the first NDEF message and other records along with their signatures from the second NDEF message. Similarly, many unrelated records along with their respective signatures can be combined together into a single NDEF message. The combined NDEF message will consist of a sequence of records that may be totally meaningless or convey misleading information, but still have valid signatures covering the whole message.

In the second scenario, the Record Composition Attack is accomplished by combining and hiding selected records from different NDEF messages. An adversary takes two or more NDEF messages signed by the same or different parties. Each NDEF message consists of records of various types like *Text*, *URI* etc. followed by the signature. The attacker takes all records from the NDEF messages and combines them to form a new NDEF message. The new NDEF message will have valid signature records corresponding to data from each parent tag. The attacker then effectively removes the unwanted records from the message by hiding them from the viewer, but keeps the signatures valid.

As all the records are digitally signed, the actual removal of any record invalidates the signature. Instead, the chosen records are retained but hidden from the user by manipulating the unsigned *TNF* field. The TNF value is changed from 1 to 5, i.e. from the NFC Forum *well-known* Type to an *Unknown* Type. The *TNF* value can be changed as it is not signed. The NDEF parser receiving an NDEF record with a *TNF* value of *Unknown* will store the payload of that record without processing it. In this case the payload will not appear to the user. So, rather than removing a record, it has been hidden simply by changing the *TNF* value.

In fact, Roland's attack [50] described thus far does not necessarily work because there are a few other changes that may have to be carried out in order to keep the signature valid. These necessary modifications were overlooked in [50]. Later, we highlighted those modifications in [62, 63]. We amended these attacks and described in detail the modifications that need to be carried.

3.1.1 Our Contribution

We amended the Record Composition Attack by highlighting some necessary revisions in the *lengths* fields that were initially overlooked in Roland's attack [50]. These amendments are described in §3.2. Our main contribution is the countermeasures we proposed to avoid Record Composition / Decomposition attacks. These countermeasures are described in §3.4.1 and §3.4.2 of this chapter. We not only published this work at international forums [63, 62], but also approached the NFC Forum to apprise them about the vulnerabilities in the signature specification and its fixes. Consequently, the NFC Forum amended the signature specification and released an updated version [9].

3.2 The Amended Record Composition Attack

For the new *TNF=5* of the hidden records, the *Type-Length* field must be zero and there can be no *Type* field (see Table 2.1). This is not the case for the original record ($TNF \neq 0, 6$). As the *Type-Length* field is not signed it can indeed be changed to zero, but the *Type* and *ID* fields are digitally signed and omitting or altering these fields to maintain a meaningful payload may invalidate the signature. Specifically, in order for the signature to remain valid, the original signature on the string $Type||ID||Payload$ has to be the same as the signature on the new string $ID'||Payload'$. These strings must therefore be identical, with its initial interpretation replaced by one with a different, possibly invalid ID' and a new, probably meaningless, message $Payload'$. Quite apart from the semantic issues, the signature verification now fails unless the number of signed bytes is the same:

$$(\textit{Type-Length})+(\textit{ID-Length})+(\textit{Payload-Length}) = (\textit{ID-Length}')+(\textit{Payload-Length}')$$

Therefore, apart from changing the *TNF* value, some further manipulation of the NDEF header may be required, together with adjustments to the *Type*-, *Payload*- and *ID*- lengths and corresponding removal, addition or repartitioning of bytes in the corresponding three fields.

When the *IL* bit is set, one easy solution is to increment the original value of the *ID-Length* or *Payload-Length* field by *Type-Length*. This corresponds to a re-location of some of the signed bytes to the *ID* and *Payload* fields. When the *IL* flag is zero this still works providing it is the *Payload-Length* field which is incremented by *Type-Length* as mentioned in [59] §V(L). It works well when the *ID* field is not present, as shown in Figure 3.1, but in the presence of an *ID* field it results in a new and probably invalid ID' field that may be detected by a semantic check. No bytes need removing or adding to the record in these cases.

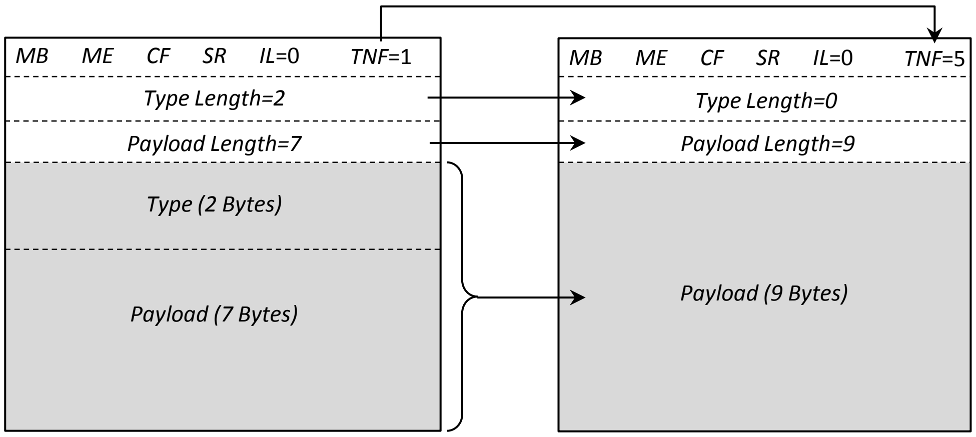

Figure 3.1: Example changes in the NDEF header when the *ID* field is absent.

We propose another solution, which cannot be caught by a semantic check on either the type field or the *ID* field. Since it has no type, the *Payload* cannot fail a semantic check either. First set the *IL* flag to zero if it is not already zero, and remove the bytes containing the *ID-Length*, as in Figure 3.2. Then increment *Payload-Length* by (*Type-Length*)+(*ID-Length*), so that the new *Payload* consists of the concatenation of all the bytes formerly in the *Type*, *ID* and *Payload* fields. In this case, *Payload′* consists of all the signed bytes.

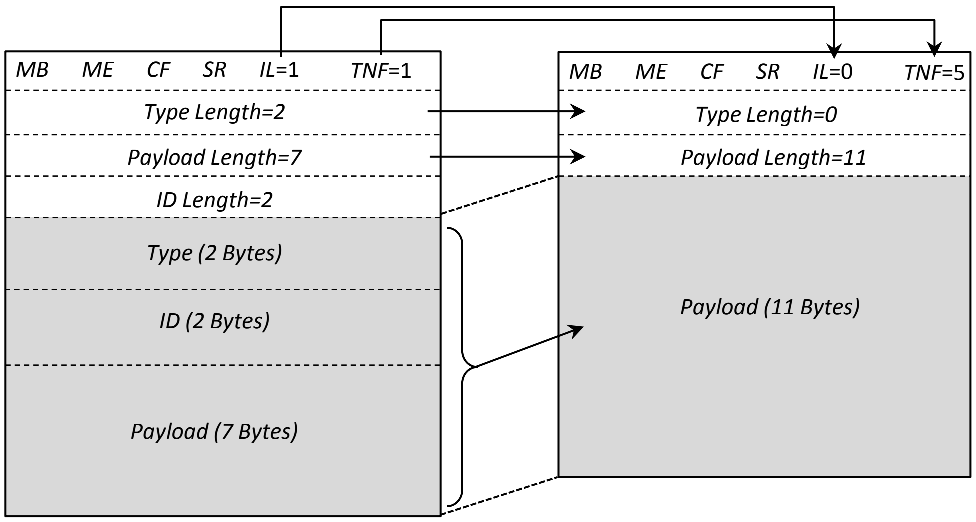

Figure 3.2: Example changes in the NDEF header when the *ID* field is present.

3.3 Record Decomposition Attack

In the second attack described by Roland *et al.* in [50], the payload is split (but *not* chunked) in two parts and spread over two records. The second part is hidden by using a record of *Unknown* type, i.e. *TNF*=5. Since *Payload-Length* is an unsigned field, it can be changed in the first record without detection. The signature is computed over the concatenated bytes from the *Type*, *ID* and *Payload* of all records being signed. So, for the two new records to generate the same signature, it just requires the second record to have no *Type* or *ID* fields. The *unknown* type does this job as a record with an *unknown* type has no *Type* or *ID* fields. The only thing required to accomplish this attack is the suitable completion of the NDEF header fields of the new *unknown*-type record.

An example of such an attack is the text of a smart poster stating, *"Do not board the train until you have a valid ticket"*. This text is digitally signed and the signature is stored using the Signature RTD. An attacker may split this message into two separate records as above. The first record stating *"Do not board the train"* will be visible to the user, whereas the second record stating *"until you have a valid ticket"* will not appear to the user as it is sent with the NDEF header fields stating *unknown* type. However, the digital signature will remain valid and so the user will consider it as a valid message. This attack of Roland works in its original form without further modification of length fields such as those described in the previous attack.

3.4 Countermeasures

Roland proposed that the receiver should only trust the payload bytes from a sequence of records if they are signed and share a common signature record [50]. But this needs very careful interpretation. As shown in the example of the Record Decomposition Attack in §3.3, the records share a common signature but only part of the message payload is displayed to the user. This partially displayed message with a valid signature cannot be trusted. Hence, the user's view of the message cannot be trusted even if all its records share a common signature.

The easiest way to avoid these attacks would be to sign all the header fields so that they may not be altered, but in practice this is out of the question. For example, the *MB* flag is intentionally unsigned so that a group of signed NDEF records may be moved to any position within an NDEF message [59]. This enables a variety of messages to be constructed for different target viewers around an important core content which is signed. However, it is unnecessary to sign the *ME* flag as the end of the section of the message which needs to have its integrity secured is marked by the signature record, so all the records preceding the signature record will have $ME = 0$.

A principle requirement of the signature definition is to be able to partition an NDEF record into multiple record chunks or *vice versa* as shown in Figure 2.5 without affecting the validity of the signature. This means, in particular, that only the *Payload* can be included in the signature for records after the first chunk, and that the chunk flag *CF* must be omitted from any header data that is included in the signature. The inclusion of any other field from the non-initial chunks, such as *length* fields, *TNF* or *CF* in the signature would also invalidate the signature. The fields from the *initial* chunk which are independent of whether or not the record is chunked are the only ones which could be included in the signature. They are the *MB*, *IL* and *TNF* fields in the header byte, the *Type-Length* and *Type* fields, and the *ID-Length* and *ID* fields.

However, one could sum the payload lengths from each chunk to obtain the same payload length as in the unchunked record, and include that in the signature because it is unaffected by chunking. This needs to be done with care as it should be possible to compute the signature using a block by block hash function without having to store every chunk payload. The message digest and total payload length must therefore be computed in parallel and, once the last chunk has been read, the length appended as a suffix to the string to being hashed. The resulting MAC is then signed and, if necessary, validated.

Another principle which we may wish to respect in proposing any revision of specifications is to insist that the signature is computed on the same components of each record irrespective of chunking. This would slightly simplify validation code, as would omitting the payload length computation.

The last principle worth mentioning is the desire to compute signatures directly from the concatenated record bytes in the order they appear and without alteration. It is easy to observe that the attacks above would not work if, for example, an extra byte B of fixed value were inserted between the *Type*, *ID* and *Payload* strings when necessary to separate them before the signature is computed. Thus, when none of the three components were the empty string, this would mean computing the signature of the string $Type\|B\|ID\|B\|Payload$, but it would be computed on just *Payload* when both *Type* and *ID* were of length 0. If this were done, the chunking process would not disturb the calculation of a signature, but the re-partitioning of bytes required in the Record Composition attack would not work. This particular solution becomes unnecessary if the lengths of the various data components are also signed.

Based on these principles, we proposed two countermeasures [62, 63].

3.4.1 Countermeasure I

This countermeasure is based on some modifications to the signature specification[1]. The signature is presently computed over *Type*, *ID* and *Payload* fields according to the signature RTD V1.0, as recalled in § 2.3.2. Because of the Record Composition / Decomposition attacks, the inclusion of additional fields is necessary. However, in order to preserve the validity of signatures when a record is chunked, a different signature process is required for non-initial chunks. The proposed modified signature is compared to the existing specification in Table 3.1.

The first byte of the NDEF header containing *MB*, *ME*, *CF*, *SR*, *IL*, *TNF* cannot be added in full to the signature as noted earlier. However, making the *TNF* value immutable is wise. Its updating was the source of problems in the Record Composition attack. So, part of the countermeasure is to sign this for all records except the non-initial chunks. For $TNF \neq 6$, create a byte *TNFB* by masking the non-*TNF* bits from the first byte of the record. This byte will be signed. *TNFB* will be the empty string for *TNF*=6, and so not alter the signature when a record is chunked.

[1]Contributed by Colin Walter

Field Name	**Existing**	**Proposed**
Message Begin *(MB)*	Not signed	Not signed
Message End *(ME)*	Not signed	Not signed
Chunk Flag *(CF)*	Not signed	Not signed
Short Record *(SR)* Flag	Not signed	Not signed
ID-Length *(IL)* Present Flag	Not signed	Not signed
Type Name Format *(TNF)*	Not signed	Signed (Unless *TNF*=6)
Type-Length	Not signed	Signed (Unless *TNF*=6)
Payload-Length	Not signed	Signed (Unless *TNF*=6)
ID-Length	Not signed	Signed (if present)
Type	Signed	Signed
ID	Signed	Signed
Payload	Signed	Signed

Table 3.1: Signing an NDEF Record

Next, we propose adding the *Type-Length* and *ID-Length* fields to the existing fields for signing except in the case of non-initial chunks, i.e. records with $TNF = 6$, when they are to be omitted. Addition of these two fields to the signature process does not invalidate the signature under the chunking process.

As noted before, *Payload-Length* cannot be signed unless the length is accumulated over all chunks. Let *Total-Payload-Length* denote the sum of Payload-Lengths over all chunks of a chunked record, and the normal *Payload-Length* for an unchunked record. For convenience, let us define *Total-Payload* similarly: it is the usual *Payload* for an unchunked record and the concatenation of the *Payloads* from all chunks of a chunked record. This means that *Total-Payload-Length* and *Total-Payload* are simply the *Payload-Length* and *Payload* of the corresponding unchunked record.

In our revised signature specification, the contribution to the signature of all chunks from a chunked record is the following string:

$$\begin{gathered}\textit{TNFB}\|\textit{Type-Length}\|\textit{ID-Length}\|\textit{Type}\|\textit{ID}\|\\ \textit{Total-Payload}\|\textit{Total-Payload-Length}\end{gathered}$$

The *TNFB*, *Type* and *ID* contributions and their lengths are, of course, those given in the initial chunk. The contribution of an unchunked record is the same, but can be written more simply as following because it is a single record.

$$\begin{gathered}\textit{TNFB}\|\textit{Type-Length}\|\textit{ID-Length}\|\textit{Type}\|\textit{ID}\|\\ \textit{Payload}\|\textit{Payload-Length}\end{gathered}$$

Because of our definitions of *Total-Payload* and *Total-Payload-Length*, the contributions to the signature are the same for an unchunked record and a chunked version of the same record. This means that the new signature could be simply defined in terms of the concatenated contributions from records in the equivalent unchunked message.

Note, finally, the ambiguity in the string used for *Total-Payload-Length*. This could be up to four bytes, and we do not know if the original unchunked record used one or four bytes if this length is under 2^8. A formal specification would have to determine how it should be given, e.g. the endianness at bit and byte level using four bytes or using the minimum number of bytes.

Suitability for the Record Chunking Process

The proposed signature scheme can be successfully used for validating messages with many record chunks without the need to store payload data. The first half of the contribution from a chunked record, namely

$$\textit{TNFB}\|\textit{Type-Length}\|\textit{ID-Length}\|\textit{Type}\|\textit{ID}$$

is wholly derived from the initial chunk. Thereafter the string for hashing is given by appending the chunk Payloads until *Total-Payload* has been appended. At the same time, a record is kept of the sum of the *Payload-Lengths* of the chunks. When the last chunk has been received, this sum equals the required *Total-Payload-Length*, and so its value can be appended also.

Therefore, a record may be partitioned into multiple chunks or *vice versa* without affecting the validity of the signature or the ease with which the signature is computed

Counter to the Record Composition Attack

The main reason for the success of the Record Composition Attack was the unsigned NDEF header fields that could be manipulated in a specific way to accomplish this attack (see Figures 3.1 and 3.2). In our proposed structure, the *TNF*, *Type-Length*, *Payload-Length* and *ID-Length* fields are signed in addition to the already signed *Type*, *ID*, and *Payload* fields. So these fields can no longer be manipulated in the required way. This makes Record Composition Attack impossible.

Specifically, in the terminology of § 3.2, for the same attack to be successful under the new signature scheme would require at least *Type-Length* = *Type-Length'* = 0 and *ID-Length* = *ID-Length'* as these both fields are signed and cannot be changed. However, *Type-Length* = 0 cannot occur when *TNF* is other than 0, 5 or 6. Although the original attack had an initial *TNF*=1 being changed to *TNF*=5, we should consider the possibility of attacks with these other initial values in order to justify (or not) the inclusion of *TNFB* in the proposed signature. This is done next.

For *TNF*=0 the record should be empty. In this case the Payload is the empty string and making it invisible will not make any difference to what the user reads. For *TNF*=5, an update to *TNF*=5 also makes no difference to the message. Finally, *TNF*=6 indicates that the record is a non-initial chunk. Changing the field to have the value 5 would change it to a non-chunked record and result in the inclusion of additional *Type-Length* and *ID-Length* fields. Although the *ID-Length* field is optional in a record, the *Type-Length* field is not. It contributes 1 byte in the new signature scheme, resulting in a different signature if *TNF* is changed from 6 to 5. We conclude that, whatever the initial value of *TNF*, updating it to 5 invalidates the signature under the proposed scheme just by virtue of including the *Type-Length* and *ID-Length* fields, no matter how the other fields are changed.

Of course, the user needs to be aware of where signed messages start and finish since any signed messages might be combined without change into a larger misleading or wrong message. The signature specification clearly defines the starting and finishing point of the data to be signed. It is up to the user's browser and security policy to make clear where signed messages begin and end. Ideally, it should show a single signed message at a time and indicate that the visible message is signed with nothing hidden.

Counter to the Record Decomposition Attack

In this attack, a record payload is decomposed into multiple parts which are completed to full (unchunked) records by the addition of relevant header fields. The *TNF* value for some parts is set to 5, making them inactive records. The header fields also contain a *Type-Length* field with value zero. As this one-byte field is digitally signed in the proposed scheme, it will contribute to the string on which the signature is computed and result in an invalid signature. The only way to avoid this byte being part of the signature is to make the second record into a chunk. However, this requires *TNF*=6 and so prevents the value *TNF*=5 which is needed to hide the record's payload in the attack. Thus, the Record Decomposition Attack is successfully countered in the proposed scheme.

The specification for the unknown type record with *TNF*=5 has some redundant data. The *Type-Length* field is always zero and therefore redundant. This redundancy, in contrast to the record chunking case, proves to be a mechanism preventing the Record Decomposition Attack. If it were removed, the heading requirement for the hidden parts of the payload in the Record Decomposition Attack would be the first NDEF header byte and the Payload-Length field. If none of this information were included in the revised signature specification, data from the header fields would not invalidate the signature. Therefore excluding this redundancy would make the Record Decomposition Attack feasible for the revised signature specification if it excluded *TNFB* and *Total-Payload*.

In conclusion, although the *Type-Length* field is redundant in an unknown type record, it helps prevent the Record Decomposing Attack. Nevertheless, other fields in the proposed signature specification ensure that this redundancy could be safely removed.

Other Malicious Combinations of Records

This section discusses other potentially malicious combinations of records with respect to the proposed signature specification. Previous analysis considered all possibilities for hiding part of a signed payload. One can ask if there are other changes to a sequence of records which would not affect the signature. The first such combination is to sign the last of the *middle* and *terminating* chunks (and perhaps more subsequent records) while omitting the *initial* and the first few *middle* chunks. Although the signature specification only covers the complete sequence of chunks, it could be abused, with the first chunk to be signed being treated as the initial chunk, contributing its values for the *ID* and *Type*, among other things. This combination might have its *Type* and *ID* properties changed since they are inherited from the *initial* chunk which may not have been signed. However, such a change is not allowable according to the NDEF specification [3]. This is because the first record in a sequence of signed records must be the first record of the message or be preceded by a signature record. As a signature record cannot be a chunk (it has *TNF*=1, not 6), the start of the signed sequence of records must be before the initial chunk. Consequently, the *Type* and *ID* properties and their lengths are always signed for the part of the payload which is signed.

Let us now consider manipulation of the unsigned bits in the first header byte. We can ignore the *MB* and *ME* flags as they do not affect the semantic content of the records.

Any alteration to the *SR* bit changes the location of the other signed bytes, such as the *Payload*. This leads to an invalid signature unless there is a corresponding addition or removal of three *Payload-Length* bytes. If this changes the value of *Payload-Length* then the signature will be incorrect as that value is signed. If that value is unchanged then the *Payload* is unchanged, so that the interpretation of the record is unchanged. Hence if the *SR* bit can be changed without invalidating the signature then the message content is unchanged.

Switching the *IL* bit without invalidating the signature is not possible except for non-initial chunks where the value is irrelevant. Moreover, the *IL* flag is always zero for non-initial chunks as defined in the NDEF specification [3]. Changing *IL* introduces a byte for *ID-Length* into the signature stream or removes it, thereby altering the signature.

Finally, we briefly comment on the need to include the *TNF* value in the signature. For example, without *TNFB*, any of the values 1, 2, 3, 4 might be inter-changed without change to the sequence of bytes being signed. This would lead to a different interpretation of the *Type* value and hence a different interpretation of the *Payload*. We would then have to rely on the parser flagging an inconsistency. It is quite possible, although unlikely, for the differently interpreted payload to convey incorrect information to the user. Thus it is still wise to include *TNFB* in the signature scheme.

Modifications in the Existing Specifications

The proposed signature scheme is different from the existing one because of the addition of, *inter alia*, *Type-Length* and *ID-Length* fields. As such, it must be assigned a different version number in order to maintain backward compatibility. Fortunately, in this specification there is a version number which can be incremented.

The payload of the signature RTD consists of three fields as noted in § 2.3.2: *Version*, *Signature* and *Certificate Chain*. These three fields are transmitted in the same order, with *Version* in first place. An NDEF parser can determine the signature specification by first analyzing the version number. This is a single byte field so it can handle up to 256 versions of signature specification. As the existing signature specification is the only version presently available, the only currently valid version number is 1. So our proposed signature specification should be given version number 2.

The proposed specification is not compatible with version 1 because of the extra signed fields. Hence signature validators will have to be upgraded to enable version 2 signatures to be checked.

Regarding NDEF specification, implementing changes in the NDEF specification is tricky as there is no *Version* field available in the NDEF format. But, our proposed scheme is designed in such a way that it does not require any alteration in the NDEF specification.

A Few Comments about the User Interface

A digitally signed NDEF message should display some information for the user at the application level. It may include the name of the signing authority (from an x.509 certificate) with some additional details for the assurance of the user.

Our consideration of security issues showed that it is still important for users to be informed whether or not messages have been signed and for their browsers to make clear where individual signed messages begin and end. It should not be possible for signed messages to be concatenated without the user being aware where one message has finished and the next has started.

A signature can potentially be removed from a tag without any indication to the user (such as in a duplicate tag). It is up to the user whether he trusts the contents of a message without a signature or not. However, it is clear from the example attacks that the browser should be pro-active in warning the user of potential dangers, including the lack of any signature.

3.4.2 Countermeasure II

We proposed another countermeasure to counter Record Composition / Decomposition Attack. The countermeasure highlights that there is some redundant data in NDEF specification. The redundant data not only results in communicational overheads but also paves the way to such attacks. We removed the data redundancy and as a result, we were able to add a few more fields to the signature record. On the positive side, this countermeasure serves two purposes; it removes the data redundancy and, in parallel, serves as a countermeasure tool against such attacks. On the negative side, this countermeasure requires revisions in the NDEF specification (in order to remove the data redundancy) and in the signature specification (as more fields are added in the signature). Whereas, the earlier proposed Countermeasure I (§ 3.4.1) requires revision only in the signature specification.

Data Redundancy in the NDEF Specification

We discovered that some of the data transmitted according to NDEF specification is redundant. The redundancy lies in the *middle* and *terminating* chunk records. The *middle* and *terminating* record chunks have *TNF*=6, indicating that the *Type* and *ID* of these records are unchanged. Therefore, the *Type-Length* and *ID-length* fields are set to zero and the *Type* and *ID* fields are omitted, as explained in §2.3.1.

The *Type-Length* field and *ID-length* fields are redundant in the *middle* and *terminating* record chunks, as *TNF*=6 indicates the same. As a result, these two fields can actually be *omitted* from the *middle* and *terminating* chunks in the revised NDEF specification.

We first revised the NDEF specification for record chunking process by omitting these two fields from the *middle* and *terminating* record chunks. The new proposed

Field Name	**Initial Chunk**	**Middle Chunks**	**Terminating Chunk**
CF	1	1	0
TNF	As required	6	6
Type-Length	Present	–	–
Payload-Length	Present	Present	Present
ID-Length	Present	–	–
Type	Present	–	–
ID	Present	–	–
Payload	Present	Present	Present

Table 3.2: Proposed construction of chunk records.

construction of a record chunk with *IL* flag set is presented in Table 3.2. The *MB*, *ME* and *SR* flags are not shown as they are used as required.

After the modifications in the record chunk structure, we proposed a revision in the signature specification. The signature in now computed over the *Type-Length*, *ID-Length*, *Type*, *ID* and *payload* fields as compared to *Type*, *ID* and *payload* in the original specification as shown in Table 3.3.

Field Name	**Existing**	**Proposed**
NDEF Header	Not signed	Not signed
Type-Length	Not signed	Signed
Payload-Length	Not signed	Not signed
ID-Length	Not signed	Signed (if present)
Type	Signed	Signed
ID	Signed	Signed
Payload	Signed	Signed

Table 3.3: Signing an NDEF Record: Countermeasure II

Counter to Record Composition / Decomposition Attacks

The Record Composition Attack (RCA) is successful because of the unsigned *TNF* value. The *TNF* is still unsigned in this approach, yet it rules out the possibility of RCA because of the inclusion of the associated fields in the signature.

The *TNF* is closely associated with the *Type* field and therefore the corresponding *Type-Length* field. For example, an NDEF record with the *TNF*=5 has no *Type* field, thus the *Type-Length* is zero. The *Type-Length* field is a single byte transmitted after the NDEF header. It always stores a non-zero integer (unless *TNF*=5 or a chunk record). This field is digitally signed in our proposed scheme and thus cannot be changed to zero. Changing the *TNF* value to 5 without changing the *Type-Length* field results in an error. Hence, the RCA is not successful in this scheme.

In Record Decomposition Attack (RDA), the payload is split in two parts and spread over two records. The second part is hidden by using a record of Unknown type, i.e. *TNF*=5. Let the attacker split the original Payload P into two parts, a visible payload (P_v) and a hidden payload (P_h). The P_v becomes the payload of the original record, whereas the P_h is stored in a new hidden record with *TNF*=5. In this way, the hidden payload will not appear to the user. However, in order to maintain the validity of the signature, the attacker must look into the *Type-Length* and *ID-Length* fields of the new hidden record, as both are now included in the signature. The attacker can omit the *ID-Length* field in the hidden record as it is optional, but the attacker needs to add a byte containing all zeros for the *Type-Length* field of the hidden record. Since this field is included in the proposed signature scheme, the new string for signature computation becomes $P_v \| 0x00 \| P_h$ which is different from the original string $P_v \| P_h$. Hence hiding records voids the signature and hence is unsuccessful.

The Record Decomposition Attack can be successful in the following, but very unlikely, conditions.

- The original payload, P, contains eight consecutive zero bits somewhere in the middle
- The string of zeros must be exactly at a location where the payload needs to split.
- Since the *Payload-Length* field represents the length of the payload in bytes, the string of zeros must encompass an entire byte.

In such circumstances, the original payload P is split into three parts, $P_v \| 0x00 \| P_h$. P_v is the payload of the visible record, $0x00$ acts as the *Type-Length* field of the hidden record and P_h is the payload of the hidden record. *ID* field, being optional, is omitted in the hidden record. In this way, the string for signature computation remains the same as it was in the original record.

3.5 Comments on the Candidate Signature Specification V2.0

The attacks mentioned above along with the countermeasures were communicated to the NFC Forum in 2012. As a consequence, the NFC Forum honoured our research and revised the current signature RTD V1.0. The NFC Forum released the candidate specification of the updated signature RTD V2.0 for comments and feedback in April 2013 [9]. The updated version V2.0 is still open for comments and not yet finalsied.

The V2.0 is not backward compatible so once implemented, the older version V1.0 will be obsolete.

Apart from a few minor changes, the major change is about the use of the signature RTD V2.0. According to the V2.0, the signature is computed over the entire NDEF record, including the first byte of NDEF record, as compared to a selection of bytes in V1.0. The first byte of NDEF record contains *MB*, *MB*, *CF*, *SR*, *IL* flags and a 3-bit *TNF* value. Although it looks more secure to include this byte in the signature, it leads to various implementation issues. These issues were highlighted by M. Roland and J. Langer prior to the release of the V1.0 [59]. We are also of the opinion that the first byte of NDEF record should not be included as a whole in the signature (Section §3.4). On the contrary, excluding this byte from signature results in various attacks such as Record Composition / Decomposition Attack. Our both proposed countermeasures handle this situation, but surprisingly, the V2.0 includes the header field in the signature. We conveyed following two issues to the NFC Forum regarding V2.0:

> *"The first problem is that the V2.0 undermines the flexibility to use multiple NDEF records within a single message. Since MB flag is signed in this approach, nothing can be added before a signed NDEF message. This prevents a variety of messages to be constructed for different target viewers around an important core content which is signed. Moreover, it is unnecessary to sign the ME flag as the end of the section of the message which needs to have its integrity secured is marked by the signature record, so all the records preceding the signature record will have ME = 0.*
>
> *The second problem with V2.0 is that the original signature gets invalid if used with chunk records. There can be many occasions when a record is required to be chunked. In such cases, the original signature gets invalid as the chunked records require a new signature. The new signature will invalidate the data origin authentication of the message unless the new signature is computed by same entity. Similarly, a signature computed on chunked records cannot be used with unchunked records. This result is loss of data origin authentication."*

3.6 Conclusion

The Record Composition and Decomposition Attacks exploit unsigned fields in the NDEF header. Previously proposed attacks were not fully implementable without further modifications to the NDEF header and *lengths* fields. We refined those attacks and explained precisely what additional changes need to be made to exploit the unsigned fields. Such attacks can be countered if the length fields of the NDEF header are also signed, but in a specific way. We proposed two solutions that require modification to the Signature RTD in which, amongst others, the *TNF*, *Type-Length*, *Payload-Length* and *ID-Length* fields are included. We presented a security analysis of both of the proposed schemes, and verified that it was no longer possible to exploit the NDEF header in attacks of the type discussed, thus successfully countering Record Composition and Decomposition Attacks in particular. We communicated our work to the NFC Forum and consequently, the NFC Forum released an updated version of the signature specification [9].

The signature specification only authenticates the static data stored on the tag, so it is vulnerable to tag cloning attacks. In the next chapter, we will design a framework that can be used for tag authentication, thus providing a mechanism to counter tag cloning.

Chapter 4

Off-line NFC Tag Authentication

This chapter provides a framework, based on NFC Forum specifications, to authenticate an NFC tag in an off-line environment. This framework is published in the International Conference for Internet Technology and Secured Transactions 2012 [64]. This chapter is almost verbatim of the published work.

4.1 Introduction

Near Field Communication (NFC) tags are used to store data in the format specified by the NFC Data Exchange Format (NDEF) specification, published by the NFC Forum [3]. Despite our investigations described in the previous chapter, here we will assume that the authenticity of the stored data is guaranteed by a digital signature which follows the Forum's Signature Record Type Definition (Signature RTD) [7]. The signature is computed over the tag's contents and is stored in the signature record on the tag along with the corresponding certificate for verification. This allows the reader to check the authenticity of the data, but the signature specification does not rule out tag cloning. The reason is that the signature specification deals only with the static data, which can be replayed or cloned. Hence, there is a need of another layer of protection for NFC tags that address cloning issues.

NFC tags are used in a variety of applications like product identification, smart posters, access control etc. There are occasions where copying the contents of an NFC tag to another tag is undesirable. An example of such a scenario is a signed NFC tag attached to a medicine packet storing its chemical composition and expiry date. Any NFC Forum device can read its contents and verify it using the signature. However, a counterfeit medicine and counterfeit tag with the same data will also be authenticated.

At present, the NFC Forum does not provide any specification to authenticate the tag. The lack of such a mechanism opens the door to many security threats, and particularly to counterfeit products when NFC technology is used in product identification.

We address this weakness by providing a mechanism based on the NDEF specification to authenticate NFC tags. The main advantage of our proposed specification is its compatibility with existing NFC Forum devices. This contribution to the NFC framework enables the successful authentication of such tags along with their data. It adds another layer of defence to the NFC security framework, making it more secure for future applications.

Tag authentication also serves as a tool for data authentication at no extra cost for a read-only tag, whereas its converse is not always true. Therefore we emphasize that tag authentication is an important security measure as it provides both tag and data authentication for read-only tags.

In an off-line environment, when there is no shared secret between the tag and the reader, it is very challenging to differentiate between legitimate and counterfeit tags (§ 4.2.2). Our framework for tag authentication is designed to work in an off-line environment. The proposal is based on a challenge-response protocol using public key cryptography and a PKI. In order to make the framework compatible with existing NFC Forum devices, a new *Tag Authentication Record*, designed according to the NFC Data Exchange Format (NDEF), is introduced.

After developing a counterfeit tag-detection framework, we also proposed its implementation architecture to detect counterfeit products in a supply chain. This will be discussed in the next chapter.

In this chapter, we require to extend the NFC Forum well-known type records, for which *TNF*=1, as our proposed *Tag Authentication Record* in § 4.4.2 falls into this category. Each of the well-known types is identified by its name, identifier and a character code as allocated by the NFC Forum. The current list of well-known types is given in Table 4.1.

4.2 Tag Authentication Scenarios

Since NFC tags can be used in a variety of applications, the techniques for tag authentication also vary a lot. Tag authentication can be categorized as follows into two main categories depending upon the tag's environment [45].

Type Name	Type ID	Hexadecimal Encoding
Generic Control	Gc	`0x4763`
Text	T	`0x54`
URI	U	`0x55`
Smart Poster	Sp	`0x5370`
Signature	Sig	`0x536967`
Tag Authentication *(Proposed)*	Ta	`0x5461`

Table 4.1: NFC Forum Well-Known Types (*i.e.* *TNF*=1)

4.2.1 On-line Authentication

In the *On-line* category, there is secret information shared between a tag and the reader as a result of the reader having access to a server containing a database of secrets. This scenario is normally applicable in a closed environment like product identification in supply chain management or access control. The reader accesses the database to obtain the tag's secret and then ascertains whether the tag knows that secret or not. Since the secret is not accessible to an attacker, a duplicate tag lacks it and can be detected.

There are various methods developed to authenticate RFID tags using an on-line authentication mechanism. Weis *et al.* proposed a mechanism to lock a tag without storing the access key on the tag [70] . Instead, the hash of the key is stored on the tag. The actual key is stored in a back end database where it can be found using the tag's ID. Unlocking a tag corresponds to tag authentication. Juels *et al.* proposed an approach to authenticate an RFID tag embedded in a bank note [41]. They provided a mechanism for law enforcement agencies to ascertain the validity of a bank note issued by a central bank. Bank notes are equipped with RFID tags and have unique serial numbers. The ciphertext on the serial number is printed on the note and stored in the RFID tag as well. The law enforcement agencies can verify the ciphertext on the serial number by communicating with the central bank. Juels [38] proposed a mutual authentication protocol for EPC Class-1 Generation-2 tags. The tag ID is altered after every authentication process to prevent traceability attacks. The cloning resistance is provided by the 'kill-password', a secret in each tag. The kill-password is unique to each EPC tag, known to tag itself and the legitimate reader. When an EPC tag receives a legitimate 'kill-password' from a reader, it self-destructs. If the 'kill-password' is not correct, the tag simply ignores the command. However, if the tag does to have sufficient power to self-destruct, it sends a 'fail' message to the reader. Hence, receiving a 'fail' message in response to the 'kill-password' is an indication that the tag is not cloned (provided that the reader has some control over the transmitted power). Ranasinghe *et al.* [57] proposed a Physical Unclonable Function (PUF) based challenge-response protocol to authenticate a tag. The challenge-response pairs, generated by a trusted party and stored on a back-end server, are used for tag authentication.

A review of existing tag authentication techniques is available in [45]. In general, these techniques use on-line servers and execute a challenge-response protocol to authenticate the tag. They cannot, in general, be adapted successfully to the off-line environment.

4.2.2 Off-line Authentication

There are occasions when there is no shared secret between the tag and the reader. Any reader can access the tag and read its contents. The process of authenticating the tag or the reader or both in such scenarios is called *Off-line* authentication. Normally, it is just the tag that needs to be authenticated. An NFC smart poster or an NFC tag for product identification falls into such a category as its contents are accessible to any reader without the need for a shared secret.

Off-line authentication becomes challenging in an RFID environment owing to the low computational power of RFID tags. The typical low cost tag is currently unable to perform any useful public key cryptography.

Pim Tuyls and Lejla Batina claim the first tag authentication model for RFIDs in an *off-line* environment [68]. They used a Physical Unclonable Function (PUF) integrated with the RFID chip (an Integrated PUF or *IPUF*). In their model, several fingerprints are derived from the PUF by sending it multiple challenges and recording the responses. The challenges and corresponding fingerprints are digitally signed and printed on the product (if used in the case of supply chain management). The verifier reads a challenge/response pair from the data printed on the product or packaging and sends the challenge to the tag to compute the response. On receipt, the verifier compares the response with the expected fingerprint. A successful match authenticates the tag.

The main drawback with this scheme is the limited number of challenges and corresponding fingerprints available in the tag's memory or printed on the product. An attacker can record all the challenge/response pairs and program another tag with the same pairs resulting in a successfully cloned tag.

Another anti-cloning approach in the RFID framework, known as *Active Authentication (AA)*, is used in ePassports where an RFID tag is used to add more security to an ordinary passport [40]. This approach uses public key cryptography where the tag digitally signs the challenge received from the reader.

Anti-cloning feature of the EMV cards provides another approach to fight cloning issues. *Dynamic Data Authentication* (DDA) verifies the legitimacy of the card by digitally signing a random challenge generated by the terminal [58]. Detail of the EMV model is described in § 4.5.3.

Alex Arbit *et al.* presented a public-key based anti-counterfeiting system for the Electronic Product Code (EPC) standard [11]. They implemented a variant of the well-known Rabin encryption scheme with a 1024-bit key [53].

4.3 NFC Tag Authentication

Tag authentication requires a framework that distinguishes a legitimate tag from a counterfeit tag. The counterfeit may or may not store the same data as the original. A duplicate with some alteration in the stored data is obviously a potential threat to the system. However, there are occasions where a duplicate with the same data is not desirable either. We describe such a tag as a *cloned* tag. Examples of such scenarios are ePassports [35], product identification, access control etc.

Conversely, there are cases where a cloned tag may be considered desirable: for example, an NFC tag used as a smart poster where the integrity of the tag contents is protected by a Signature record. The more a smart poster is cloned, the more its contents are advertised.

As observed in § 4.2.2, NFC tags may be used for product identification in an *off-line* environment. The information stored in the NFC tag is product specific and aimed to assist an off-line user to know about the specification and legitimacy of the attached product. The data on the tag is protected by the signature record and a valid signature is an indication of a genuine product. Unfortunately, the same data can be stored on a duplicate NFC tag affixed to an inferior product. The signature remains valid as it is the same data as in the original tag. Since the signature is valid, the user is led to believe that the product is genuine, whereas it is not. This happens because the signature specification authenticates only the stored data on NFC tag. An easy way to avoid such attacks is to include the tag's ID in the signature in order to detect a cloned tag with a different tag ID. But an attacker can affix to the counterfeit product a programmable tag which returns the same ID as the original. In this case, the counterfeit is authenticated as a genuine product with very little investment by the attacker.

This attack works because the tag is not authenticated along with its data and a static identifier is used to authenticate the tag. Lack of any tag authenticating mechanism opens doors to counterfeit products being accepted as genuine products.

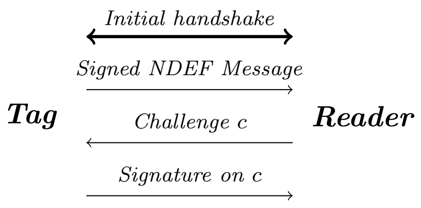

Figure 4.1: Proposed Tag Authentication Protocol

At present, the NFC Forum has not specified any mechanism to authenticate the tag. In the absence of such a mechanism, NFC tags based on the NFC Forum specification cannot be used for secure product identification.

In this chapter, we propose a solution to detect cloned NFC tags used in an *off-line* environment. Our solution is formulated within the NDEF specification and introduces a new *Tag Authentication Record* containing parts of a digital certificate. The main advantage of introducing such a new record is its compatibility with existing NFC Forum devices. The authentication is then performed using a challenge-response protocol employing public key cryptography and a PKI.

4.4 Proposed Tag Authentication Scheme

As noted in § 4.2, NFC tags can be used in both *on-line* and *off-line* environments. Our scheme for authenticating a tag in an *off-line* environment is based on the *Active Authentication* scheme of the ePassport [35] but modified to fit the NFC architecture. The scheme is applicable to at least NFC Type-4 tags as these tags are computationally powerful enough. Such tags satisfy ISO 7816-4 and contain a cryptographic processor. They can compute asymmetric or symmetric key encryption [2]. The scheme may be applied to other tags in future as their computational power increases over time. Moreover, light-weight versions of public key encryption schemes may also appear and allow wider applicability. In the scheme, the tag signs a challenge and the reader verifies the signed response as shown in Fig. 4.1.

The following assumptions are required for the scheme to work:

- Both the reader and the tag can perform public key encryption/decryption.
- The memory location where the secret key is stored inside the tag is not accessible to any reader.
- The reader possesses the root CA certificate to validate the signature.

Our scheme differs from the standard methods of authentication used in smart cards/SIM cards because the latter requires an on-line environment. A SIM card stores a secret key K at a secure location in the card. The same key is also stored with the mobile network operator. During authentication, the network executes a challenge-response protocol to verify knowledge of K by the SIM. A cloned SIM should lack the key and so would be detected. A similar approach is used in smart cards issued by banks for monetary transactions. However, in our context there is no opportunity to share a secret. Although it is feasible with the right equipment to 'hack' the NFC tag and recover the private key, we will assume that it is not cost-effective for an attacker to do this and that therefore the private key is not present in the cloned tag,

4.4.1 The Tag Authentication Digital Certificate

Before describing the protocol, it is useful to define the public key certificate which it uses and the structure of the proposed new *Tag Authentication record* which stores part of its contents. The certificate requires at least the following six fields:

1. *Protocol Version*
2. *Challenge Signature Scheme*
3. *Challenge Public Key*
4. *Supplementary Text*
5. *Certificate Signature Algorithm*
6. *Certificate Signature*

The *Protocol Version* field determines which version of the *Tag Authentication* specification is being used. This allows for future expansion and developments. The *Challenge Signature Scheme* field specifies which digital signature algorithm, along with the relevant parameters, is used by the tag to sign the challenge, and the *Challenge Public Key* field stores the public key information associated with verifying this signature. The *Certificate Signature* field is the signature on the records containing the first four fields, computed using the algorithm defined by the *Certificate Signature Algorithm* field. It follows the NFC Forum signature specification scheme [7]. The *Supplementary Text* field has various uses which are described below.

The certificate might follow the X.509 specification [6], or a simplification with fewer critical fields. Clearly, it may be necessary to add further fields in future, such as an expiry date to deter the illegal re-use of tags. In the case of X.509, the *Extensions* field enables the inclusion of the *Supplementary Text*. That field also allows the certificate to be restricted to this NFC application, which is useful because less computationally extensive cryptography is expected than is normally acceptable in other situations. We will not consider the encoding of the certificate any further beyond observing that space is at a premium on a typical tag. Therefore one would want to reduce, for example, the twenty or so bytes used for identification of the signature algorithm in an X.509 certificate to just one byte.

The stored data and data transmitted between tag and reader is in the format of NDEF records. In order to store the public key information, we propose a new *Tag Authentication record*. The first three certificate fields are to be stored in the *payload* field of this record, the *Supplementary Text* is stored in normal NFC Forum text records on the tag, and the last two fields are placed in an NFC signature record.

4.4.2 The Tag Authentication Record

The *Tag Authentication Record* serves two purposes: it stores the public key information necessary to verify the signature on a challenge, and its presence is a claim that the tag is equipped with anti-cloning features. Indeed, such records should not be allowed in tags without the ability to enact the authentication protocol. The proposed record follows the NDEF structure given in Fig. 2.2 with a *TNF* value of 1. Its *Type* should therefore be added to the set of NFC Forum well-known types, and the type name of 'Ta' (for 'Tag Authentication'), with hexadecimal encoding 0x5461, added to the list in Table 4.1. Given the general nature of the construction, the record may have much wider uses in future, authenticating other entities than tags. It may therefore make more sense to call it a *Certificate Record* with a different type identifier.

4.4.3 The Supplementary Text Field

As space is limited on tags, it will often be useful to have a single signature covering some or all of the message content in the tag, rather than having separate signatures on the certificate and the message for users. The *Supplementary Text* field enables this to be done by using it for message content. The signature then ties this content to the particular tag. For convenience, it is assumed in the implementation details below that one signature is indeed used to cover both the text message and the *Ta* record fields.

In future, the *Supplementary Text* may have to be structured into subfields if it is used for several purposes.

4.4.4 Protocol Execution Sequence

The scheme is executed in two phases:

1. **Initialisation Phase.** After selecting the version number and signature scheme, a public-private key pair (K_{pub}, K_{pr}) is generated in a secure way by a trusted party and the private key K_{pr} is stored inside the tag at a secure location only accessible to the tag processor for prescribed operations. The public key K_{pub} is stored in the *Challenge Public Key* field of the payload of the authentication record. This *Ta* record, along with the other records stored on the tag, is then digitally signed by the same trusted party and the signature is stored in the *signature record* on the tag according to the NFC Forum Signature Specification. This turns the *Tag Authentication record* into a signed digital certificate applicable to NFC tags.

2. **Verification Phase.** The verification phase is executed as follows (see Fig. 4.2):

   - The reader requests and obtains all the data from the tag. This data is in the form of NDEF records.
   - The reader verifies the integrity of the tag's contents by verifying the signature. A valid signature indicates that the tag contents are authentic.
   - Next, the reader looks for a *Tag Authentication record* by searching for record type 'Ta'. Its absence indicates that the tag is not protected by the anti-cloning feature and the Tag Authentication protocol terminates unsuccessfully.
   - Otherwise, the *payload* of the *Tag Authentication record* is extracted and, assuming the reader can support the required signature scheme, an appropriate challenge c is generated and sent to the tag.
   - The tag now uses its private key K_{pr} to compute the digital signature of c with the specified padding, and sends the result back to the reader.
   - The reader verifies the signature on c using the public key K_{pub} from the certificate. Successful verification indicates knowledge of the secret key and hence the legitimacy of the tag, whereas failure indicates a cloned or damaged tag.

4.5 Analysis

This scheme successfully detects a cloned tag from an original tag because a cloned tag lacks the private key K_{pr} corresponding to the public key K_{pub} available in the *Tag*

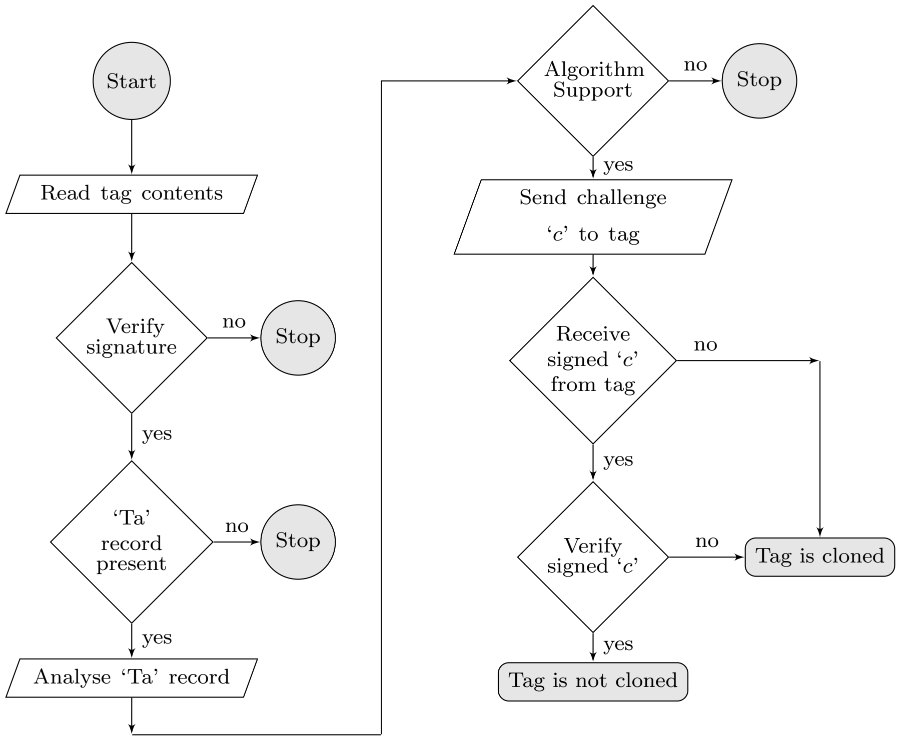

Figure 4.2: Verification Process

Authentication record. However, there are several issues that need to be discussed in detail.

4.5.1 Backward Compatibility

The backward compatibility of the tag authenticating protocol can be assessed by following two scenarios. We use "plain" to describe a reader without authenticating ability, and a tag without authentication response ability.

- **An Authenticating reader and a plain tag.** In this case, the reader starts the verification process as shown in Fig. 4.2. The plain tag lacks this feature so there should be no *Ta* record in its contents. In this case the verification process stops with the conclusion that the tag cannot be authenticated. However, if there is a *Ta* record, then the plain tag is unable to respond to the challenge. Again the process terminates with the conclusion that the tag cannot be authenticated. Since a plain tag should not contain a *Ta* record, the reader can reasonably conclude that the tag is cloned. Notice that this requires that plain tags are not loaded with a *Ta* record. As we noted earlier, existence of the *Ta* record should be viewed as a claim that the tag *does* support the authentication protocol.

- **A Plain reader and an authenticatable tag.** In this case, the *Ta* record is present in the tag but the reader does not support the tag authentication protocol. Although the reader recognises the *TNF* field value 1, the record type 'Ta' is unknown to it. According to the NFC Forum specification [3] §3.2.10, an NDEF parser receiving an NDEF record with a supported *TNF* value but an unrecognised *Type* field must interpret that record as being of *Unknown* type, i.e. *TNF*=5. So the *Ta* record is ignored. Thus the system remains backward compatible in this scenario as well.

4.5.2 Implementation Issues

One of our concerns in our proposal is the implementation of public key cryptography on NFC tags. We have not evaluated our protocol for any particular public key algorithm, but the chip area, current consumption and clock cycles are important factors to consider alongside the security of the algorithm. So the selection of the algorithm that can be implemented on an NFC tag is of prime importance. Some lightweight public key algorithms are being proposed, notable among them is the WIPR, proposed by Yossef Oren and Martin Feldhofer [52]. WIPR offers 80 bits security and fits completely (including RAM) into 5705 Gate Equivalents (GE) and has a mean current consumption of 10.88 μA with 66048 clock cycles. WIPR is more efficient than ECC-192 but it uses much more resources than AES (3400 GE with 3.4 μA and 1032 clock cycles) as shown in Figure 4.3[1].

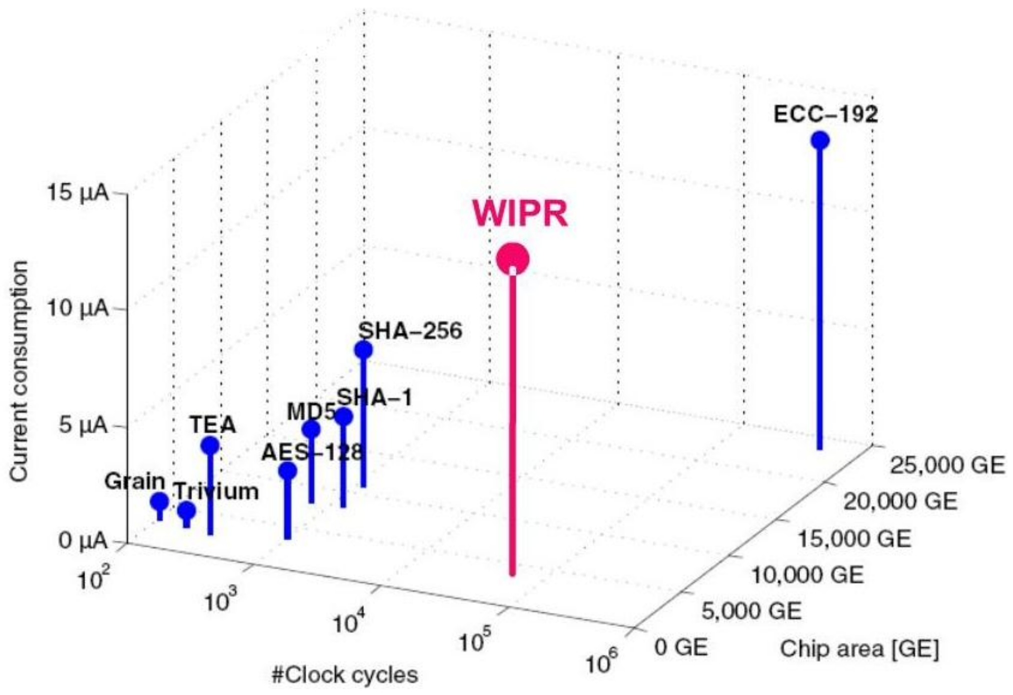

Figure 4.3: Comparison of WIPR with other Algorithms (Based on Table 4 in [53]).

4.5.3 EMV Card Authentication Model

The EMV standard defines three card authentication mechanisms [58, 25]:

1. **Static Data Authentication (SDA)**: SDA is an off-line authentication method where the card provides a digitally signed data (e.g. card number, expiry date) to the terminal. the terminal knows the issuer's public key, it can authenticate the data stored on the tag. Due to the static nature of the authentication data, the same set is used in all transactions throughout a card's lifetime. Moreover, the data is transmitted in clear to the terminal during authentication, hence easily intercepted by adversaries while in transit. These weaknesses allow replay attacks or card cloning attacks.

2. **Dynamic Data Authentication (DDA)**: DDA is implemented on the cards that support public key cryptography and have a public/private key pair. DDA

[1]The figure is presented by Martin Feldhofer and Yossef Oren on a talk at Second ACM conference in Wireless Network Security (WiSec) [53]. http://www.docstoc.com/docs/170423767/WIPR-WiSec

verifies that a card is genuine by verifying the existence of a valid card resident cryptographic key. This is carried out by a challenge-response mechanism, where the card proves its authenticity by signing a challenge chosen by the terminal using a private asymmetric key. Unlike SDA, this does rule out cloning and replay attacks.

After the authentication of the card, DDA does not tie the subsequent transaction to the card.

3. **Combined Data Authentication (CDA)**: CDA repairs this deficiency of DDA. With CDA, the card digitally signs all important transaction data, not only authenticating the card, but also authenticating the transaction.

Since all three authentication mechanisms require a Public Key Infrastructure to function, the EMV has standardized it [24]. For SDA cards, the card issuer's private key is used to generate a digital signature on the card data. The issuer (e.g., bank) then puts the digital signature and the corresponding CA-signed issuer's certificate onto the card. Each ATM/POS terminal has the actual CA public key available, and hence can verify the legitimacy of the issuer's certificate. The terminal then verifies the signature on the card's data.

For DDA and CDA, the card issuer generates a public/private key pair for each card. The private key is stored inside the card at a location only accessible to card processor. The corresponding public key is stored in a card certificate, and the card issuer (e.g, bank) signs the card certificate with its private key. The card certificate is stored on the card together with the issuer's certificate. The issuer's certificate is signed by the CA and each ATM/POS terminal has the actual CA public key. Thus, every certificate in the certificate chain is verified with the verification of the signature.

A closer look on these mechanisms reveals that the SDA is analogous to the NFC Forum Signature specification (explained in Chapter 2, § 2.3.2), whereas the DDA is analogous to our proposed Tag Authentication specification (described in this chapter). The NFC Forum signature specification provides a guideline to digitally sign the tag's contents, but it does not prevent tag cloning. The proposed Tag Authentication Record is based on a similar model to DDA and rules out the cloning attacks.

4.5.4 Tag Message as a Digital Certificate

It was noted above that the *Ta* record and the rest of the tag message can be signed separately, giving two signature records, or can be signed as one, yielding a single signature record. The former provides a clean separation between the tag authentication processes and the message authentication processes at all levels from signing to verification. However, the latter binds the message to the specific tag: the message cannot be ported to another tag because that tag does not know K_{pr}, and will be detected as noted above even on a tag which does not have authentication capability. This solution also seems preferable because of restricted space. Respectively, these two alternatives both have interpretations of the *Ta* record with its corresponding signature record and optional message as a digital certificate.

4.5.5 Strengths and Weaknesses

The strength of our proposed scheme relies on the strength of the signature scheme, secure location of the private key K_{pr} inside the tag's memory and the integrity of the public certificate verification key K_{cert}, say, in the reader. The first is a well matured area of information security, and the emergence of elliptic curve cryptography means that at least the certificate fields on the tag can be signed securely to yield a fairly short signature. However, the cryptography used by the tag to sign the challenge will often be fairly weak because of the low electrical or cryptographic power available to the tag. One needs to recognise that an attacker can send numerous challenges to the tag and record the replies and may therefore be able to break a weak system. The second issue, the maintained secrecy of K_{pr}, depends very much on what the customer is prepared to pay for the tag, and accessibility to the tag during its life. Unless the attacker can recover K_{pr}, he is unable to use the tag's digital certificate on a clone. It is indeed easy for an attacker to extract the key from a cheap tag if it can be taken to a laboratory for further study. Thirdly, if the integrity of the key K_{cert} can be compromised, the attacker can make a successful clone of a tag even when it is protected by a *Ta* record. The attacker stores his own private key K'_{pr} in the cloned tag and the corresponding public key K'_{pub} is stored in its *Ta* record, which the attacker signs. The public verification key K'_{cert} corresponding to this signature is then used to replace the correct key in the reader. Consequently, the reader believes it has an authentic tag when it verifies the clone. This results in a successful attack. This attack can be avoided if the verification process also includes verification of the certificate chain stored in the Signature record. The certificate chain does not store the top-most certificate (e.g., the root Certificate Authority certificate in an X.509 certificate chain), therefore the cell phone needs to access it online, or through any other mean, if the root certificate is not already stored in its memory. Additionally, the cell phone also needs to check the validity of all certificates in the certificate chain for any revocation by visiting the Certificate Revocation List (CRL).

In our proposed scheme, the integrity of the *Ta* record is assured by digitally signing this record, and perhaps others, according to the signature specification (Version 1.0) provided by the NFC Forum [7]. Recent attacks on signed NDEF records [63] put a question mark on the integrity of the tag's contents even if they are signed. The *Ta* record can be made inactive in a cloned tag by changing its *TNF* value to 5 with some compensating alteration in the length fields, as mentioned in [63], to preserve the validity of the signature. Since the *TNF* and length fields are not included in the signature, these alterations will not invalidate the signature. Now the verifier will not execute the tag authentication protocol since it does not recognise the *Ta* record as such. Nevertheless, if the reader is expecting an authenticatable tag, it should flag this to the operator [2].

4.6 Conclusion

Application of NFC technology to monetary and similar transactions requires strict adherence to appropriately sound measures to ensure the necessary high level of security in the NFC framework. The recently published NFC Forum Signature Specification provides assurance of data integrity in NFC tags through the digital signing of NDEF messages. However, no mechanism is provided by the NFC Forum for detecting a counterfeit or cloned tag. This results in various possibilities for malicious activities where a legitimate tag is replaced by a counterfeit tag and the NFC tag reader is unable to detect the counterfeit. We proposed a framework to counter such attacks by providing a tag authenticating mechanism, "analogous to the EMV's DDA on bank cards". It introduces a new *Tag Authentication Record* that provides relevant information to authenticate a tag in an *off-line* environment. It employs public key cryptography with digital certificates and so can be used on NFC tags that have sufficient computational power and resources to perform such operations. The *Tag Authentication Record* is based on the NFC Data Exchange Format and is thus compatible with all NFC Forum devices. The NFC tag simply signs a challenge c and returns the signature to the NFC reader. The NFC reader verifies the signature according to the information available in the previously communicated *Tag Authentication* record. A successful verification confirms that the tag is not cloned. Of course, the certificate chain should also be checked for any revocation.

In the next chapter, we will apply this framework in a supply chain to detect counterfeit products. The legitimacy of products, equipped with NFC tags, will be determined by authenticating the NFC tags; a cloned tag implies a counterfeit product.

[2]This referenced vulnerability of the signature specification is addressed by adding more fields to the signature in the NFC Forum revised signature specification Version 2.0

Chapter 5

Consumer-Level Counterfeit Detection Framework

In this chapter we present a framework to detect counterfeit products in a supply chain using NFC tags. This work is published in the International conference on Privacy, Security and Trust (PST 2013) [61]. This chapter is almost verbatim of the published work.

5.1 Introduction

Our tag authentication model, proposed in the previous chapter, can be used to detect counterfeit products in a supply chain when products are equipped with clone-resistant NFC tags. A cloned tag points to a counterfeit product. This chapter provides a counterfeit detection model where consumers can detect a cloned tag (or in other words, a counterfeit product) with their cell phones.

Counterfeit products are one of the major threats to modern commerce. According to estimates by the Counterfeiting Intelligence Bureau (CIB) of the International Chamber of Commerce (ICC), counterfeit goods make up 5 to 7% of world trade [13]. Counterfeits are available in a wide range of products, typically starting from high value small goods like watches, designer clothes, DVDs and electronic chips to high cost items such as cars, motorcycles and bicycles.

Counterfeit products are classified into four categories [16].

1. The first category consists of those products that are inexpensive, lower quality and may lack original packaging. This category is often called 'knockoff'. These products are being sold as counterfeits and the consumer is aware of it.

2. In the second category of counterfeit, a genuine product is reverse engineered and identical copies are sold as the genuine product. It is hard for a consumer to differentiate between a genuine and a counterfeit product. This category is meant to deceive the consumer.

3. These are the products that are produced by an outsourced manufacturer without intimation to, and without permission from, the original owner. For example, an outsourced manufacturer manufactures further product after termination of its contract with the original owner without notifying the original owner.

4. These are genuine products that do not meet the manufacturer's standards but are not labelled as faulty.

One of the major outlets for counterfeit products is Internet e-commerce where the consumer has no means of authenticating a product before delivery. Even after delivery, the consumer has very limited resources to determine the legitimacy of a product. Auction websites, such as eBay, have further expanded the market of counterfeit products. For example, test purchases from 300,000 Dior products and 150,000 Vuitton items offered on eBay during 2006 found 90% counterfeits [49]. Tiffany & Co. purchased 186 random items from eBay and found only 5% to be genuine [20].

These circumstances call for mechanisms to fight counterfeiting. Analysis shows that the money spent in this way prevented a much greater loss from counterfeit goods. According to the U.S. Chamber of Commerce, \$5 is gained for every \$1 invested in this battle [67].

Radio Frequency Identification (RFID) tags attached to various goods provide a tool to remotely identify these goods. Among these, EPC tags are the most important. The Electronic Product Code (EPC) network is used for supply chain management and can be used as a tool for anti-counterfeiting [65]. Every item equipped with an EPC tag carries a 96-bit code to uniquely identify and manage the item in a supply chain. There are two main approaches to using the EPC as an anti-counterfeiting measure [44]. The first approach is tracking the physical location of a tag and updating the result in an online database. The EPC of a counterfeit product will appear twice (at least) in the database, assuming the counterfeit product is equipped with a cloned EPC tag and the database is up to date. This is called the 'Track and Trace' approach. The main disadvantage of this approach is its significant communication and computation overheads. Every reader has to update records in the online server in real time. The online server has to track and trace each code received from the online reader and generate triggers in case of any abnormality. In addition to these overheads, there are also some privacy concerns associated with this approach: for example, tracking of individuals from the products they carry, or tracking medicines etc [12]. Moreover, there are some issues with updating the database. For example, suppose a retailer were to clone the EPC and attach the cloned tag to a counterfeit product. Assume he does not update the corresponding record in the database when it is sold. When he sells the counterfeit product, the consumer buying the counterfeit can check and find a valid record for the cloned EPC but will not be able to update its record to record its sale due to limited access and privacy concerns.

Another anti-counterfeiting approach is based on cryptography. In this approach, each tag contains a secret value, knowledge of which is established by the reader in an authentication proof. Generally, this uses an encrypted challenge-response protocol as it may be eavesdropped and the secret cloned if sent in the clear. This approach may be based on symmetric key or asymmetric key cryptography.

Anti-counterfeiting based on cryptography can be categorized into two main categories as described in § 4.2: *Off-line* and *On-line*. In a supply chain, it is very unlikely that the login credentials are provided to a consumer to access the database in order to verify the authenticity a product. There could be a special login account just for verification with restriction (no update or general read rights). But this may lead to information leakage if not properly implemented or monitored. This makes the *Off-line* approach more suitable for product authentication at the consumer level.

If symmetric key cryptography is used, the reader must already know the secret value of the tag and match it against the secret value received from the tag. The secret value of each tag is chosen uniquely so that if a tag is compromised, it should not break the entire system. This results in a requirement for a secure and efficient key distribution mechanism to distribute the tags' secrets among the readers. One way is to deliver the secret values of all appropriate tags to readers in advance but this approach requires secure distribution of millions of such keys and is considered infeasible. Another way is to store the key database in an online server. This server is online at all times to provide the secret values of tags to readers. Assuming millions of tags are deployed in the supply chain with hundreds of compatible RFID readers, this approach incurs even higher communication and storage overheads than the track and trace approach [44]. In addition, the reader must always be trusted by the supplier since the reader stores the secret values of the tags in any framework employing symmetric key cryptography.

As observed earlier, one of the major factors in the upsurge in counterfeit products is online shopping. With the advancement in Internet technology, the volume of online shopping is growing rapidly. It is not feasible at present to tailor any symmetric key approach for product authentication to online shopping. The reason is obvious: a consumer receiving a product through online shopping does not possess an RFID reader to communicate with the tag attached to the product. Even in the very unlikely scenario where a consumer possesses an RFID compatible reader, the supplier will have to provide login credentials to access the database. This situation is far from practical. Thus, product authentication at the consumer level remains an open challenge, especially for the Internet shopping framework.

In contrast to the symmetric key approach, asymmetric key cryptography (or Public Key Cryptography (PKC)) can also be used to authenticate a product. Considering the limitations of the symmetric key approach described above, the case for PKC in product authentication is thus very strong. The main restriction in using PKC on RFID tags, such as EPC tags, is the limited computational and storage capabilities of these tags. Recently, Alex Arbit *et al.* presented a working implementation of a PKC-based anti-counterfeiting framework [12]. We revised this model and provided a better approach to counterfeit detection.

5.1.1 Our Contribution

We focus our work on detecting counterfeits that fall into the categories 2 and 3 mentioned in § 5.1. Category 1 counterfeits are not a major concern as the consumers are aware of the fact that the products they are buying are counterfeits. The loss in the sales of the original product owner is also negligible as very few genuine goods purchasers would purchase a knock off [16]. Category 4 counterfeits can be restricted by the genuine product owner by enforcing efficient quality control measures. Categories 2 and 3 are most critical as not only is the consumer unaware of the illegitimacy of the product, but also the genuine owner has no or minimal control over the production, marketing and selling of such products. Our model helps in detecting counterfeits products at consumer level pertaining to category 2 and 3 products, thus providing an efficient tool to detect counterfeits.

In this chapter, we analyse the anti-counterfeiting model which Alex Arbit *et al.* proposed in [12] and highlight a few of its short-comings. The framework is semi-offline, where the verification and decryption keys are dispatched to the reader using a smart card and the reader is considered as a secure module for storing these keys. Its semi-offline structure renders it sometimes incapable of authenticating a product at consumer level despite using public key cryptography.

We revise and extend their work in two main ways. Firstly, we restore the EPC tag to the original standard rather than using the modified EPC tag in the Alex Arbit *et al.* model. This resolves any modification-related problems in the existing EPC framework. Secondly, we supplement the EPC tag with an NFC tag which can perform the necessary computations that were not within the capability of the EPC tag. The main advantage of being offline is that a consumer can authenticate a product without any online communication with the supplier's database. Message integrity in our proposal is provided by digital signatures so the consumer needs to verify the certificate chain for the validity of the certificates. This can be done by connecting to the Internet and checking the Certificate Revocation List (CRL). We believe that our offline product authentication at consumer level can prove to be an efficient anti-counterfeiting tool. Although our framework is applicable to all levels of supply chain management, the main beneficiaries are customers using the Internet for online shopping. This framework not only helps the customers to authenticate a product, but any verifier such as a law enforcement agency can also use this model to detect counterfeit products.

We resolve the problem of provisioning of an RFID reader for product authentication to every consumer by also using a Near Field Communication (NFC) tag for the EPC. NFC technology is now available on cell phones and so a consumer's cell phone can act like an RFID reader to read the EPC. Since our framework is totally offline, the consumer is able to distinguish between a legitimate and a counterfeit product by using his cell phone without accessing the supplier's database.

We also resolve the issue of trust in the reader for an offline framework. In the work of Alex Arbit *et al*, the reader is a secure module storing a verification key and a decryption key, as noted earlier in this section. These keys cannot be stored on any reader that is untrusted by the supplier. Although the consumer's cell phone is not trusted by the supplier, this issue can be addressed by using a Public Key Infrastructure (PKI), thereby all but eliminating any key storage requirement on the reader side. In many cases, the NFC tag can also be accessed and authenticated during product distribution without having to resort to the greater reading range of the EPC tag.

5.2 EPC in the Supply Chain

The EPCglobal Class-1 Gen-2 (*EPC C1G2*) standard [5] specifies low-cost UHF tags which operate in the frequency range of 860-960 MHz and have a read range of 2-10 meters. This longer range makes UHF tags more easily read in containers and warehouses than is the case with NFC tags. Electronic Product Code (EPC) tags are typically deployed in supply chain management systems for automated inventory checks. The EPC is a 96-bit identifier stored in the EPC tag which helps to identify each tagged product uniquely. EPC has various advantages over existing product identification techniques, e.g. barcodes, as the former does not require line of sight compared to the latter. Moreover, the EPC tag may store additional information about the product which cannot be achieved using a barcode. Because of these advantages, barcodes are often being replaced by EPC tags in the supply chain.

5.3 Related Work

The EPC network as an anti-counterfeiting tool was proposed by Staake *et al.* [65]. The proposal is based on a central database server and does not explicitly cover the use of cryptography. The BRIDGE project [44] analysed various anti-counterfeiting approaches based on RFID tags. This work analysed the secure distribution and management of secret keys in a symmetric key anti-counterfeiting framework, and showed that it results in ten times more communication and computational overheads than in a track-and-trace anti-counterfeiting system.

Work to reduce the computational overheads in public key cryptography is also in progress and various lightweight public key cryptosystems are being designed. The CRYPTOGPS is a light-weight public-key cryptosystem mainly suitable for UHF RFID tags. It can be implemented in around 2800 GEs (Gate Equivalent) with a processing time of around 720 cycles [55, 19]. The Rabin Cryptosystem was the first to be implemented in a wireless sensor network in [53]. It took about 16,700 GEs to implement 512-bit encryption. This led the authors to declare that this cryptosystem was unsuitable for resource-constrained RFID tags. A lower version of this scheme, WIPR, was developed by Oren and Feldhofer [11]. It is well suited to RFIDs because it has the smallest hardware footprint and largest payload capacity of all published high-security public key schemes. Recently, Alex Arbit *et al.* presented a working implementation of a WIPR based anti-counterfeiting framework [12].

5.3.1 The Alex Arbit *et al* Anti-Counterfeiting Model

Alex Arbit *et al* proposed an anti-counterfeit model based on EPC tags and Public Key Cryptography [12]. Their framework is illustrated in Figure 5.1.

The figure represents the various entities involved in the anti-counterfeiting framework. The framework consists of the following sequence of operations.

- **Step 1:** The framework is initiated by the Tag Integrator, who wishes to deploy anti-counterfeiting technology in EPC tags. He creates two public-private key pairs: a Private Signing Key K_S together with its Pubic Verification Key K_V, and a Private Decryption Key K_D with its Public Encryption Key K_E. The signing key K_S is never disclosed to any entity of the framework. The Tag Integrator generates a list of Tag Identifiers (TIDs) and signs each TID with the

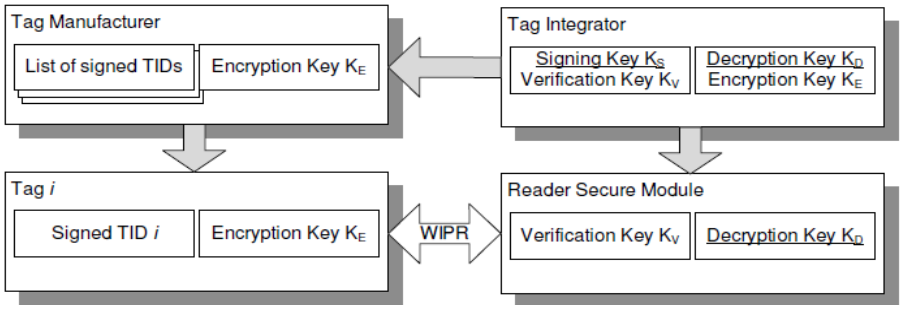

Figure 5.1: Alex *et al* anti-counterfeiting framework

key K_S. He then sends the list of signed TIDs to the tag manufacturer along with the encryption key K_E. Since the tag manufacturer lacks K_S, he is unable to generate arbitrary signed TIDs, thus ensuring the integrity of the TIDs.

- **Step 2:** The tag manufacturer produces and deploys the tags, each with an individually signed TID from the list along with the public key K_E.
- **Step 3:** The reader receives K_D and K_V from the tag integrator. Once these keys are delivered to the reader, the system can operate in an offline framework. The reader then carries out a challenge-response protocol to determine that the tag possesses a valid, signed TID.

This is a semi-offline model as it requires an initial key distribution mechanism to distribute keys to readers through some secure channel. The authors suggest distributing keys through a secure module such as a smart card.

5.3.2 Weaknesses

There are several weaknesses[1] in this model.

- The framework is semi-offline where the reader stores K_V and K_D. This puts a limit on its utility for product authentication at consumer level, as K_D cannot be communicated to the consumer.
- K_V and K_D have to be delivered to a reader through some secure channel such as a smart card. Since the same set of keys are distributed to each reader, this results in a single point of failure where the loss of a single smart card will compromise the entire system. Moreover, if a single retailer is dishonest, he can break the entire system as all the readers use the same set of keys K_V and K_D.

[1]Contributed by Zeeshan Bilal, ISG, RHUL

- The authors have not discussed the storage location and accessibility of K_E inside an EPC tag. If K_E is stored at an accessible location, an attacker can make a successful counterfeit tag by simply copying all the content of the EPC tag, including K_E, to a counterfeit tag. If K_E is stored at some inaccessible location inside the EPC tag, it can prevent tag cloning, but still the framework is prone to single point of failure. Since K_E is identical in each tag, it only needs an adversary to attack a single tag to compromise the entire system.

- **Bypass Attack.** The framework is prone to a "Bypass" attack where the anti-counterfeiting protocol is circumvented in a counterfeit tag in the following way. The framework is designed to handle both WIPR-modified and standard EPC tags. During the handshake protocol between a reader and an EPC tag, the tag responds with an indication of being WIPR capable or not. This is achieved by the modified tag sending a special WIPR EPC message to the reader instead of the actual EPC value according to the standard (see Figure 4 in [12]). The special WIPR message acts as a flag to the reader to execute the anti-counterfeit protocol.

  The framework does not provide integrity protection to the special WIPR EPC message contents and so alterations to this message may not be detected. An attacker just needs to replace this message with the actual EPC value in the counterfeit tag, thereby making the tag claim to follow the standard EPC protocol. On receipt of the actual EPC value from a counterfeit tag, the reader does not execute the anti-counterfeiting protocol, instead assuming the tag to be unmodified as the flag (the special WIPR message) is not received from the tag. Thus, the anti-counterfeit protocol is bypassed and the counterfeit tag remains undetected. Of course, if the reader knows the TID belongs to a tag which follows the WIPR modified protocol, then the counterfeit should be detected.

5.4 The Proposed Model

In this section, we propose a different anti-counterfeiting model that uses RFID technology to detect counterfeit products. This model is a modified version of the Alex Arbit *et al.* model [12]. We add an NFC chip to the EPC tag, thereby providing a product authentication mechanism to the consumer level.

NFC technology is used mainly for two reasons. Firstly, this technology can support public key cryptography on tags and, secondly, it is available on cell phones enabling them to act as RFID readers. The former supports our framework in an offline mode

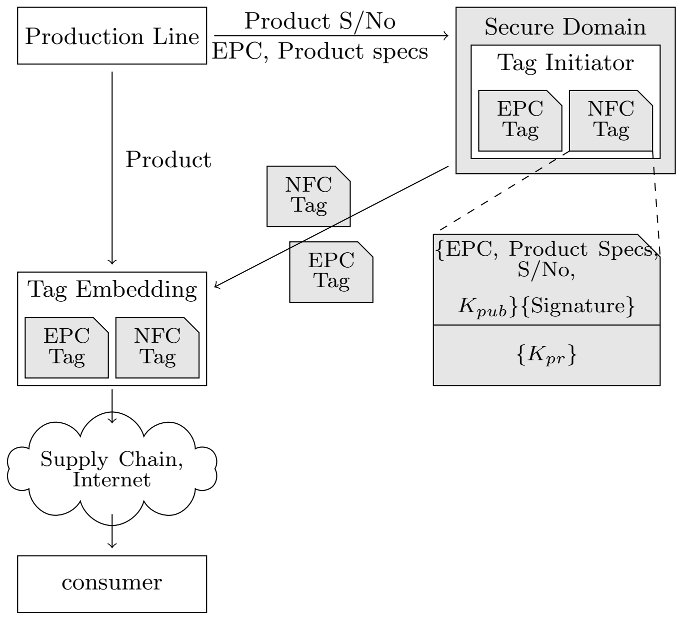

Figure 5.2: Initialisation phase of the proposed scheme.

where a connection to the supplier's database is no longer required. The latter helps extend the authentication model to the consumer level, where a consumer uses his cell phone to authenticate a product.

5.4.1 Initialisation Phase

Our new anti-counterfeit model is executed in two phases, the first, namely initialization, being illustrated in Fig. 5.2. This phase is initiated from the production line where a serial number and an EPC are allocated to the product. The serial number, EPC and the product specification are communicated to Tag Initiator *(TI)*. Meanwhile, the product is dispatched to the Tag Embedding department.

On receipt of the information from the production line, the *TI* generates a public/private key pair (K_{pub}, K_{pr}). This pair is unique for each tagged product item. The *TI* must be a secure platform as it is responsible for the generation of anti-counterfeit keys. It stores the EPC in an EPC tag and forms a string S_1 defined by

$$S_1\text{=EPC}\|\text{Product S/N}\|\text{Product Specification}\|K_{pub}$$

The *TI* digitally signs this string S_1 with his signing key K_{sign} and stores the string along with its signature on the NFC tag. The signature on the tag is stored as a 'Signature Record' according to NFC Forum's Signature Record Type Definition [7].

According to this specification, the signature record consists of a digital signature along with a digital certificate containing the corresponding verification key K_{ver} with a hierarchical certificate chain as described in Chapter 2 §2.3.2. S_1 and its signature are stored at a memory location accessible to any NFC reader. However, the *TI* also stores the secret key K_{pr} inside the tag but at a secure location. This location of K_{pr} is only accessible to the tag's processor and therefore inaccessible to a reader. The corresponding public key K_{pub} is a part of S_1, and therefore accessible to any NFC reader. After storing the relevant information on both tags, the *TI* configures both tags as write protected and dispatches them to the Tag Embedding department.

On receipt of the tags from the *TI*, the Tag Embedding department embeds both tags on the product. Since the tags are physically embedded we shall assume that any attempt to remove the tags will destroy them. After embedding the tags, the products are shipped to the supply line, from whence they may be delivered to a department store or direct to a consumer through online shopping.

5.4.2 Verification Phase

This phase is executed by the verifier on receipt of the product. Since this is an offline framework, the verifier does not require any connection to the supplier's database. Therefore the verifier may be a consumer, a warehouse employee, a member of law enforcement or indeed, any individual wishing to authenticate the product. The verification phase is executed in two phases. The first is visual and the second is cryptographic. The visual verification process is executed as follows:

- The consumer checks the claimed identity of the product itself and the integrity of the tag which should be bound to the product item in a tamper-evident manner.
- The verifier places his cell phone on the NFC tag to read its contents. The accessible data on the NFC tag (string S_1 and corresponding signature) is communicated to the cell phone.
- The cell phone verifies the signature. A successful verification is an indication that the string S_1 is legitimate.
- The cell phone displays the product specification and its serial number to the consumer.
- The consumer checks the two product descriptions match each other.

In the case of a successful visual verification, the consumer should initiate the second phase of product verification, which is a cryptographic challenge-response protocol:

- The cell phone sends a random challenge r to the NFC tag.
- The tag signs r with the secret key K_{pr} and returns the result $sign(r)$ to the cell phone.
- The cell phone verifies the signature using the corresponding verification key K_{pub} which it knows from S_1.

A successful verification is a strong indication of a genuine product, as a counterfeit tag lacks the signing key K_{pr} and so cannot compute a valid signature on r.

The verification process also includes verification of the certificate chain stored in the Signature record. The certificate chain does not store the top-most certificate (e.g., the root Certificate Authority certificate in an X.509 certificate chain), therefore the cell phone needs to access it online, or through any other mean, if the root certificate is not already stored in its memory. Additionally, the cell phone also needs to check the validity of all certificates in the certificate chain for any revocation by visiting the Certificate Revocation List (CRL).

5.5 Analysis

In this section, we analyse the proposed framework from various angles. Our model addresses category 2 and 3 of counterfeits as mentioned in § 5.1.1. Categories 1 and 4 are not a focus of our work since, in the former case, the products are being identified by the consumers as counterfeits and, in the latter, can be countered with an appropriate quality control. Categories 2 and 3 are critical as the consumer is not aware of the illegitimacy of the product. Since our model is designed to detect counterfeits at the consumer level, it provides a tool for consumers to determine the legitimacy of a product.

In the case of category 2 counterfeits where the original product is reverse engineered, the NFC tag attached to the original product cannot be reverse engineered – the secret data on the NFC tag cannot be copied as explained in § 4.5.5 and § 5.5.2. A consumer can therefore determine the illegitimacy of a reverse-engineered product by the unsuccessful verification of the data on the NFC tag.

In our model, the Tag Initiator (TI) is responsible for generating and storing the secret keys on the NFC tags. The tags are then embedded on the product by another department termed the 'Tag Embedding Department'. In the case of out-sourced manufacturers, the product manufacturing and tag embedding are done by the out-sourced manufacturer. The TI remains a part of the genuine owner and is not out-sourced or, if it is, it is to a trusted partner only. The genuine owner provides NFC and EPC tags to the out-sourced manufacturer only in same quantity as specified in the contract. If an out-sourced manufacturer is dishonest and produces more than the quantity mentioned in the contract (category 3 counterfeits), he will have to produce the product either without the NFC tag or with a fake NFC tag. This counterfeit product is then detected by the consumer because making a fake NFC tag is too difficult (explained in § 5.5.2). Of course, the consumer needs to be aware that the product is equipped with an NFC tag and reject the product if it is absent. Thus, our model helps in the detection of category 3 counterfeits at consumer level.

5.5.1 Justification for Two RFID Tags

We use two types of tags in our framework, an EPC tag and an NFC tag. Although both are RFID tags, they have very different characteristics. The main difference is the operating frequency: EPC tags operate at 860-960 MHz whereas NFC tags operate at the 13.56 MHz frequency band. The range is consequently different in the two tags. EPC tags can be read from 2 to 8 meters whereas NFC tags have a very short communication range of no more than about 4 cm. This property makes only the EPC tag suitable for supply chain management in order to remotely identify the products. Since EPC tags are already deployed in the market for supply chain management, we use EPC tags in our framework in order to maintain the backward compatibility and normal supply chain needs.

NFC tags are used because two main requirements cannot be fulfilled by EPC tags. Firstly, EPC tags are very resource constrained when compared to NFC tags: EPC tags have very limited computational power and much less memory, whereas NFC tags, especially NFC Type-4 tags, are much more powerful. Since our framework is based on PKC and PKI, where the tag has to perform PKC, we need a reasonably well-resourced tag. Secondly, our framework needs to provide authentication down to the consumer level. Without an NFC tag, this would require every consumer to be equipped with an EPC tag reader, which is currently far from practical. The issue is resolved with the inclusion of the NFC tag, as the consumer's cell phone can act as a reader for the tag.

5.5.2 Security Analysis

In this section, we analyse our framework from the security point of view. The goal of an attacker is to develop a clone tag or a tag with a valid signature. To develop a clone tag, the attacker must know the private key K_{pr} of the original tag. This key is stored at an inaccessible location in the tag's memory and so it is normally secure from the attacker. The alternative solution open to an attacker with a cloned tag is to replace the legitimate public key K_{pub} with the attacker's public key K'_{pub} in S_1 and store the corresponding private key K'_{pr} in the tag. However, this is not possible as the legitimate K_{pub} is digitally signed (it is in a digital certificate) so that any alteration will invalidate the signature. Of course, the verifier must have a trusted source for the certificate's verification key in order not to be duped. Since our framework is based on Public Key Infrastructure (PKI), it inherits the complexity issues, such as certificate management, certificate revocation etc of the PKI. For instance, the user needs to access the Certificate Revocation List online whenever it needs to verify a certificate chain.

In case an attacker spends time and money to reverse engineer or penetrate a single tag and recover its private key K_{pr}, it will not affect the entire system as the pair K_{pr}, K_{pub} is unique to each tag. The tags, being cheap, will have few counter-measures to side channel analysis, which will be a sufficient threat in some markets. However, this will avoid a single point of failure as experienced in Alex *et al* model.

Our framework is resistant to the bypass attack. The existence of K_{pub} in S_1 is an indication that the tag is equipped with the anti-cloning feature. This key can neither be removed nor altered as it is digitally signed. The user's application on the cell phone, once it has detected K_{pub}, will execute the anti-counterfeiting protocol, thereby resisting the bypass attack.

In addition to cryptographic authentication, our framework also provides visual product authentication. After scanning the NFC tag, the product specification and product serial number is visually displayed on the user's cell phone display. The user can visually check and verify the information from the product or product packaging. Needless to say, there are many other sources of compromise. For example, the NFC tag could just return a QR code which connects the consumer's phone to the attacker's website and displays the expected protocol output and the verification data for the counterfeit product. Alternatively, the merchant may direct customers who lack the verification app to the attacker's website to download a compromised app that confirms the authenticity of any product.

The tags have to be tamper-evident. This is to ensure that they cannot be re-used on counterfeit products. If the tag were to contain the URL for registering the product under the manufacturer's guarantee, customers could be encouraged by their app to register, the manufacturer could check its database for duplicate registrations that would flag a clone, and the manufacturer could advise the consumer if there were such a problem.

One critical factor in securing the system is the physical location of the NFC tag in the product. This is an industry specific decision and requires careful consideration. It is assumed that the tags are physically embedded on the main assembly of the product and not on casing/packing or on any easily replaceable component of the product, very much in the same way as a watermark or hologram is an integral part of the item it is protecting. As in the latter case, an attacker just needs to place the tag embedded component from a legitimate product into the counterfeit product.

5.5.3 Mobile Phone Security

Mobile phones are never considered as trusted platforms. They are vulnerable to various types of viruses, worms, trojans, rootkits and botnets [42]. They are also used as attacking platforms for data skimming, relays attacks and card emulators. This implies that a compromised mobile device may not give the desired output in our scheme. For instance, a malicious application running on a compromised mobile device in our scheme may always display a genuine product of some specific brand, irrespective of the legitimacy of the product. Additionally, the customer must download the product verification application from a trustworthy source. Otherwise, there are chances that a malware gets downloaded in the mobile device undermining the product authentication mechanism. Needless to say that the genuine downloaded application must be protected against known attacks.

5.5.4 Economic Analysis

This section analyses economic aspects of the proposed scheme at a broader level.

The inclusion of NFC tags in addition to EPC tags for product identification requires some additional investment by the supplier. For simplicity, we assume the additional costs associated with generating keys, signing certificates, writing to the tags, embedding the tags, etc., is already included in the cost of the NFC tag. We also assume that the cost of an item is independent of the number of such items made, which is plainly rather naïve.

We only consider loss in the sales revenue because of counterfeit products. The true loss is much higher and not just financial. There are various other important aspects such as loss in distinctiveness of brand image, gradual decline in sales, unemployment etc., but inclusion of these factors complicates the analysis too much – our goal is merely to justify the cost of our anti-counterfeiting scheme.

Let x be the production cost/unit and y the selling price/unit, $\Delta = y - x$ the profit/unit, n the market demand over some fixed period and p_i the initial percentage of counterfeits in the market (i.e. prior to implementation of our framework).

We assume that our scheme does not fully eliminate counterfeits from the market. This assumption is based on two factors. Firstly, not all the consumers will touch their mobile phones with NFC tags to determine the legitimacy of the product. Some may not even know about it, and some may ignore this procedure. Secondly, a compromised mobile device may not detect a counterfeit product as explained in §5.5.3. So, we expect that after implementation of our scheme by a product manufacturer, some counterfeit products will still be in the market. We denote the percentage of counterfeit product after implementation of our scheme as final percentage, p_f.

Similarly, we denote the initial financial gain G_i and final financial gain G_f for a company before and after implementation of our framework respectively.

Suppose, by observing his sales, the original manufacturer is able to make exactly the number of products he can sell, namely $n(1 - p_i)$. The remaining market share of np_i consists of counterfeits from other suppliers. The G_i by the original manufacturer is $n \cdot \Delta \cdot (1 - p_i)$ compared with an ideal profit of $n \cdot \Delta$ if he were to supply the whole market.

Let c be the unit cost of implementing RFID tags on a product. This cost includes all associated costs regarding RFID implementation as mentioned earlier. If no price increase is allowed, the final financial gain G_f generated under these conditions is:

$$G_f = n \cdot (\Delta - c)(1 - p_f)$$

This represents an increase providing $G_f > G_i$, i.e.

$$n \cdot (\Delta - c)(1 - p_f) > n \cdot \Delta \cdot (1 - p_i) \tag{5.1}$$

The initial percentage p_i of counterfeit products depends on various factors like brand, geographical location, in-store or on-line, product price etc. It is difficult for any agency, other than the product manufacturer itself, to find an exact value of p_i for a specific brand as the counterfeit products of categories 2, 3 and 4 are indistinguishable. This may be the reason behind the large variation in the estimate of counterfeit products by the Counterfeiting Intelligence Bureau of the International Chamber of Commerce (7% counterfeits), and the surveys carried out by some product manufacturers like Dior, Vuitton, Tiffany & Co., on eBay (about 90% counterfeits) as described in § 5.1. Assuming both these figures to be true, they illustrate that the percentage of the counterfeits products p_i varies from very low level to high level.

Similarly, the value of p_f is hard to predict with a reasonable accuracy. We claim that our scheme will decrease the percentage of counterfeits, but by how much? The answer depends on various factors like p_i, consumer awareness, availability of a smart phones and corresponding mobile phone application to the consumers, and most importantly, the methodology of determining p_f by a brand. For simplicity, we say that the $p_f = m \cdot p_i$, where $0 \leq m \leq 1$. To get a rough estimate of the economic feasibility of our model, we analyse it under following four values of m:

- $m = 0.25$ (counterfeits are decreased by 75%)
- $m = 0.50$ (counterfeits are decreased by 50%)
- $m = 0.75$ (counterfeits are decreased by 25%)
- $m = 0.90$ (counterfeits are decreased only by 10%)

Assuming the price of implementing RFID tags with the required infrastructure is \$5/unit, the Eq (5.1) can be written as:

$$\Delta > 5\left(\frac{1-p_f}{p_i-p_f}\right) \tag{5.2}$$

The plot of Equation (5.2) is shown in figure 5.3 with above-mentioned four different values of m. The shaded region in the figure represents the suitability of trying to prevent counterfeits using our model.

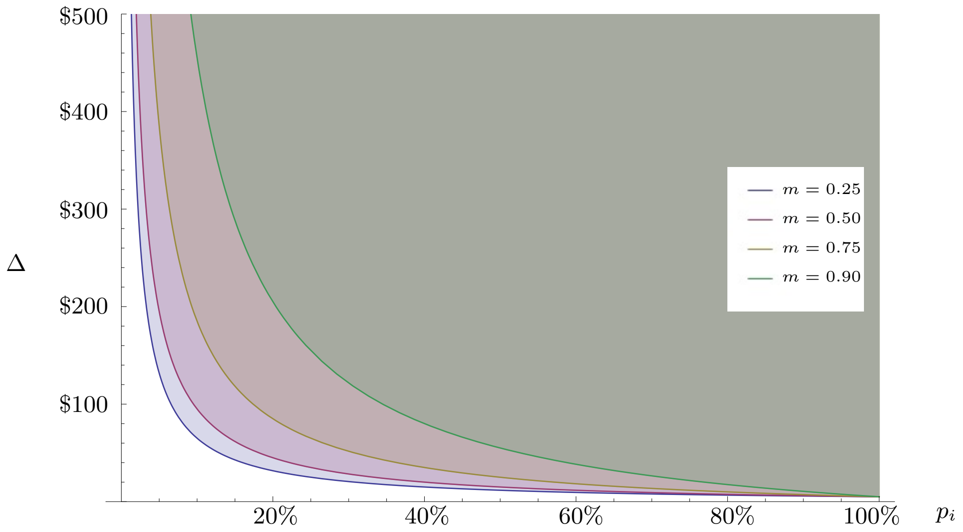

Figure 5.3: Suitability of Economic Model ($c = \$5$)

Table 5.1 describes the minimum profit/unit (rounded to nearest Dollar) for the extreme values of p_i (7% and 90%) against the four assumed values of m.

This analysis is based on very straightforward assumptions and may not represent the reality. For instance, we assumed that the product cost is not related to the number

m	$p_i = 7\%$	$p_i = 90\%$
0.25	\$94	\$6
0.50	\$138	\$6
0.75	\$271	\$7
0.90	\$670	\$11

Table 5.1: Minimum Profit/Unit, Δ, above which our model is likely to be feasible

of items being produced, which is far away from practicality. Similarly, loses apart from sales revenue, for example, damaged brand value and firm reputation, health and safety risks due to sub-standard products and its concerning financial effects, loses in Foreign Direct Investment (FDI), cost of legal actions taken against counterfeiters and pirates, cost on investigations to detect counterfeits in a market, etc., are also not considered in our model. Moreover, there may not be a linear relationship between p_f and p_i in a practical scenario. Addition of these factors in economic analysis makes it more complicated, that we don't want to – our goal is only to focus on the financial effects of our model at a broader level. This simple analysis represents that our model becomes more suitable for the markets facing high number of counterfeits. For a market with low percentage of counterfeits, our model is suitable for only high cost items. On the other hand, for a market with high percentage of counterfeits, our model is suitable for even low cost products. Although product manufacturers are very much aware of profit/unit of their products; they must have a reasonable data of counterfeit products in the market in order to decide about implementation of our model.

5.6 Conclusion

This paper presents an RFID based anti-counterfeiting framework at the consumer level. There are two main constraints related to this authentication level. Firstly, the typical individual consumer cannot afford to keep an RFID reader to authenticate a product; and secondly, customers cannot be provided with access to the supplier's database because of privacy issues. We addressed both these constraints by using NFC technology: an NFC tag is used along with an EPC tag for consumer level authentication on the reasonable assumption that most individuals will carry an NFC-enabled mobile phone in the near future. We provided a dual layer verification mechanism to a consumer. In the first phase of verification, the product specifications are displayed to the consumer on his cell phone for visual verification against the actual product. After successful verification, a cryptographic challenge-response protocol is executed to authenticate the product. Our proposal is based in PKC and PKI and successfully detects the counterfeit products. Analysis shows that the proposed framework is likely to be feasible for low-cost products with a profit/unit above about \$6 for the markets facing high volume of counterfeits, whereas, our model is suitable for only high cost product (profit/unit above \$94) for the markets with fewer counterfeits under some straightforward assumptions. These values may not represent the reality, as many other side-effects of counterfeits are not considered, just to keep our analysis simple.

Once a tag is registered against a specific user , then tag authentication leads to user authentication. In the next chapter, we will describe a payment framework, where the payer is authenticated from the SIM card followed by a monetary transaction using the mobile device.

Chapter 6

An NFC Payment Framework

This chapter provides a mobile transaction framework. The work is published in the Eighth International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM) [60]. §6.4 and §6.5 are almost verbatim of the published work. §6.2 is contributed by Pardis Pourghomi and Gheorghita Ghinea (both from Brunel University, Uxbridge, Middlesex).

6.1 Introduction

One of the main use of NFC technology in future is mobile payments. At a broader level, the mobile payment comprises of two parts: user authentication and transaction execution. NFC, being a relatively new technology, has yet to mature in this domain as still there is no standardised framework for monetary transaction using NFC. In this chapter, we propose an NFC based payment solution. The Mobile Network Operator (MNO) of the user performs the transaction on behalf of the user. The MNO first authenticates the user, and after successful authentication, the MNO performs transaction. The authentication of the user is carried out through the mobile device, which follows a different approach from the authentication frameworks described in earlier chapters of the thesis. The reason is the existence of a shared secret key K_i between the mobile device and the MNO. This is an online tag authentication (described in Chapter 4, §4.2.1) which complements our earlier proposed off-line authentication framework in chapter 4. We used GSM authentication parameters in our approach not only to authenticate the user, but also to encrypt the communication channel. Our work in this chapter is a step towards making NFC a tool for mobile commerce in future.

6.2 Mobile Commerce Using NFC

Mobile Commerce, also known as *m-commerce*, is the ability to conduct commerce using a mobile device, such as a mobile phone, a Personal Digital Assistant (PDA), a smartphone, or other emerging mobile equipment such as dashtop mobile devices. The use of m-commerce has seen rapid growth in recent years, with several different services like Short Message Service (SMS), Wireless Application Protocol (WAP), Unstructured Supplementary Service Data (USSD) and K-Java on the GSM network and NFC [21].

NFC technology over mobile devices has given a new direction to m-commerce. With NFC technology, mobile phones can have additional functionality to act as a contactless card to be used as an easy method of payment. However, there are concerns such as personalization, data storage, management and ownership of the Secure Element (SE) as it stores sensitive data such as banking credentials. The trust among various key players, such as MNO, mobile device manufacturers, banks, SIM manufacturers etc, is the key for a successful mobile transaction framework. NFC, being a relatively new technology, is yet to mature to be widely used in m-commerce.

Alpár *et al.* introduced Tap2 technology where the users need only their NFC-enabled mobile devices and credentials implemented on their smart cards [10]. They proposed the use of NFC technology in the on-line banking solution based on EMV Chip Authentication Program (EMV-CAP). Gerald Madlmayr and Josef Langer presented a purse-based micro-payment system [46]. They designed a pre-paid wallet where the money is stored in the Secure Element in the mobile device. The user can top-up their account Over-The-Air (OTA), anywhere and anytime.

W. Chen *et al.* proposed an authentication and transaction protocol that utilizes the existing GSM network for customer authentication and monetary transaction [21]. The protocol first authenticates a customer who wants to pay for some services. He uses his mobile phone for payment and the respective MNO transfers funds from his account to the shop account. The same researchers proposed another transaction protocol that combines the existing 3G cryptographic primitives and algorithms, in addition to the identification and authentication of the customer, with the NFC technology to implement a mobile payment system [22].

Inspired by this approach, we proposed two separate solutions with a similar pattern, i.e., the MNO is responsible for transferring funds from a customer to a shop. We name these versions Protocol Version I and Version II.

We will describe some salient features of the Version I in § 6.3. This will be followed by its improved version, Version II in § 6.4.

Before going into the details of our work, we would like to provide an overview of

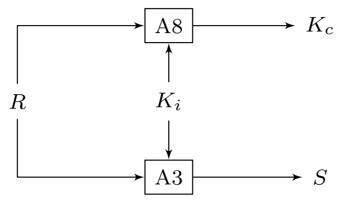

Figure 6.1: Generation of K_c and S from R.

the GSM authentication and advantages to a cloud-based SE.

6.2.1 GSM Authentication

When a mobile device signs into a network, the Mobile Network Operator (MNO) first authenticates the device (specifically the SIM). The authentication stage verifies the identity and validity of the SIM and ensures that the subscriber has authorized access to the network. The Authentication Centre (AuC) of the MNO is responsible for authenticating each SIM that attempts to connect to the GSM core network through a Mobile Switching Centre (MSC). The AuC stores two encryption algorithms, A3 and A8, as well as a list of all subscriber identities along with their corresponding secret keys K_i. The key K_i is also stored in the SIM. The AuC first generates a random number, denoted by R. This is used to generate two responses: a signed response S and a key K_c as shown in Figure 6.1, where $S = E_{A3,K_i}(R)$ uses the A3 encryption algorithm and $K_c = E_{A8,K_i}(R)$ uses the A8 encryption algorithm [1].

(R, S, K_c) is known as the *Authentication triplet* generated by the AuC. The AuC sends this triplet to the MSC. On receiving a triplet, the MSC forwards its first element R to the mobile device. The mobile device SIM computes the expected response S from R, using A3 and the key K_i which is stored in the SIM. The mobile device transmits S to the MSC. If this S matches the S in the triplet, then the mobile is authenticated. K_c is then used for communication encryption between the mobile device and the Base Station (BS).

6.2.2 Conventional Payment Structures

A conventional payment structure consist of following main entities (as shown in Figure 6.2). The role of each entity is described in [29].

- **Card and Card Holder:** It is the end product user; the one who possesses a payment card and to whom the card is issued.

- **Merchant:** It is the entity which accepts payments from a card holder in exchange for goods and/or services and connects to a payment network through an acquirer.
- **Acquirer:** It is a third-party service provider that acquires and processes payment transactions for merchants, manages the relationship with the global and regional payment networks on the merchants behalf and manages the transaction database. The acquirer connects merchant transactions to payment networks by:
  - Providing the POS device to the merchants
  - Securely routing transaction from POS device to the payment network
  - Managing transactions from authorization to clearing to settlement.
- **Issuer:** This is the financial institution which issues a card to a cardholder and holds the account or credit line behind the card. It performs many activities that could include:
  - Cardholder customer service
  - Data preparation
  - Configuration set-up
  - Fulfilment of personalized chip card, with all paper inserts; preparation for mailing to customer
  - Define card profile, including risk parameters
  - Receive and manage card records and keys to form a personalization record
  - Generate personalization script
  - Key management activities for EMV, CVV/CVC, and PINs between card manufacturer and personalization bureau and between issuer and personalization bureau

6.2.3 Advantages of the Cloud-Based Approach

Our NFC cloud-based approach is based on managing and accessing sensitive transaction data by storing the data in a cloud rather than in the mobile phone. When a transaction is carried out, the required data is retrieved from a remote virtual SE which is stored within the cloud environment. The mobile phone SE provides temporary storage and authentication assets for the transaction to take place.

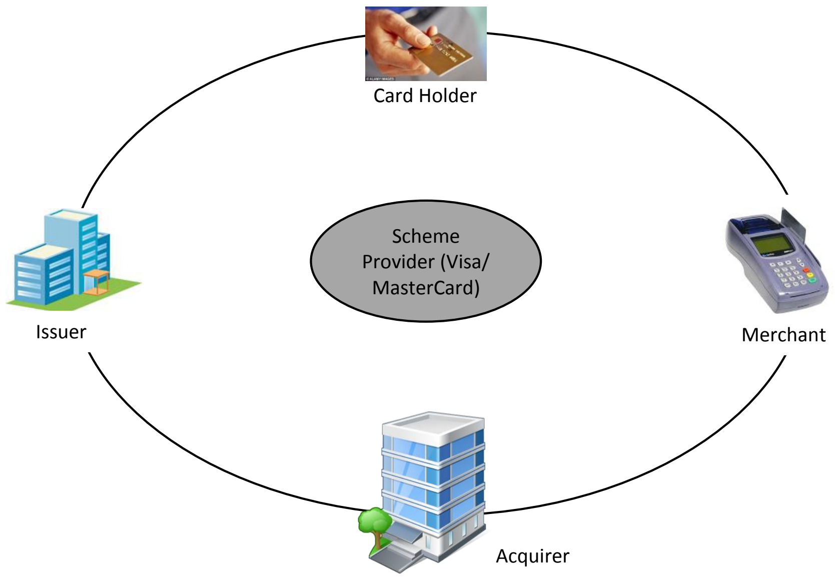

Figure 6.2: A conventional Payment Structure

An issue with SEs is that companies have to meet the requirements of organizations such as EMVco to provide high level security in order to store personal data [54]. This makes the SE expensive for companies. However, a cloud-based approach would transfer this cost. Then the SE in the NFC phone is only responsible for user/device authentication and not for storing personal data. This improves the cost efficiency of the SE compared with the present, enabling many more secure applications to be supported because of the reduced pressure on space. Also, the NFC controller chips could be smaller and cheaper as they no longer have to support all previous functionality.

The NFC cloud-based approach makes business simpler for companies in terms of the integration of SE card provisioning. It would be much easier for businesses to implement NFC services without having to perform card provisioning for every single SE. The NFC phone user will be able to access many more applications as they are no longer stored in a physical SE. In terms of flexibility, all users would be able to access all their applications from all their devices (e.g. phones, tablets or laptops) since the applications are stored in a cloud environment that provides a single, shared, secure, storage space. Moreover, fraud detection would be instantaneous as the system runs only in a fully online mode.

6.3 The Protocol Version I

The Version uses a cloud architecture where the cloud is being managed by the MNO. The MNO first authenticates a customer through an improved an improved GSM authentication and after a successful authentication, transfers the required amount to the shop. Its main features include:

- Mutual authentication of the mobile device and the MNO using existing GSM primitives
- PIN verification
- Multiple accounts against a single customer
- *Pay-as-you-go* and *pre-paid* accounts for payments
- Non-repudiation of transaction messages by using digital signatures

This work is published in the International Journal of Advanced Computer Science and Applications (IJACSA) [56].

6.4 The Protocol Version II

The major improvement in the Version II is the elimination of the shared secret between the shop PoS terminal and the customer MNO, a prerequisite in the Version I. Therefore, the shop does not need to get itself registered with the customer MNO to perform mobile transactions. This makes the Version II more flexible and it can even be used for monetary transfer between two individuals provided that the payer has registered an account with his MNO.

Additionally, we also eliminate the requirement of secure channels among various entities of the MNO. We suggest a dedicated department, MNO Transaction Department (MTD) to manage the monetary transactions. The user provides his bank card details to the MTD for *pay-as-you-go* transactions, or top-up his account for *pre-paid* transactions. The registered bank card is not required to be physically present during transaction, so our model falls under *Card Not Present (CNP)* category. The MTD in our model is analogous to acquirer in conventional payment structure. It acts as a third party responsible for monetary transaction. The issuer is the financial entity (e.g., bank) that issues bank card to card holders for *pay-as-you-go* transactions, whereas, for *pre-paid* transactions, it is subsumed by the MTD. The latter facilitates those individuals who do not have their bank accounts but they need to pay for services. A virtual secure tunnel is established between the mobile device and the MTD to ensure the security of the messages. The virtual tunnel is of special significance when the Base Station of some other network is used for the transaction; as in such scenario, the MNO responsible for monetary transaction does not want to reveal any sensitive information to the Base Station.

The Version II has many similarities with the Version I. The Secure Element (SE) is partitioned into two sections; one stored in the SIM for customer authentication and the other stored in a cloud, managed by the MTD, to hold customer's account details. A customer, who is a user of a cell phone, opens up a payment account with the respective MNO prior to use the proposed payment feature. Each account is identified by a unique identity, the *Account ID* or Acc_{ID}. The account is either a *pre-paid* or a *pay-as-you-go* account. A customer can have one or more accounts of either type and has the option to select one account while payment.

In contrast to the Version I, the mobile device communicates with the MNO over the standard GSM link. The shop communicates with the customer MNO through the customer's mobile device using NFC and the GSM link. Communication over the GSM link between the mobile device and the Base Station is encrypted as specified in the GSM standard. Otherwise, communication between different entities of the GSM network is not considered to be secure and so encryption needs to be added where appropriate. The MNO may be linked to the customer's mobile device through its own BS or through a BS of some other network. Especially in the latter case, the proposed protocol should not disclose any sensitive information to the other network. The shop PoS terminal does not require to be registered with the MNO. This makes monetary transactions between two individuals possible (the payer and the payee are analogous to the customer and the shop respectively).

The customer's cell phone is equipped with an application installed in the SIM that provides all required functionalities and a user interface for transactions. For simplicity, we refer to the cell phone and the SIM as a single unit, the 'Mobile Device' (MD). The authentication of the customer by the MNO is derived from the GSM authentication.

The shop PoS terminal and the customer's MD, both need to obtain and store trusted certificates for the keys of the MTDs they are willing to trust. K_{sign}, K_{ver} are the signing and verification keys respectively of the MTD, whereas K_{pr}, K_{pub} are the private decryption and public encryption keys respectively of the MTD.

The protocol executes in three different phases: customer identification and credit check, customer authentication, and transaction execution. Steps of the protocol are illustrated in Fig. 6.3, with numbering as in the text.

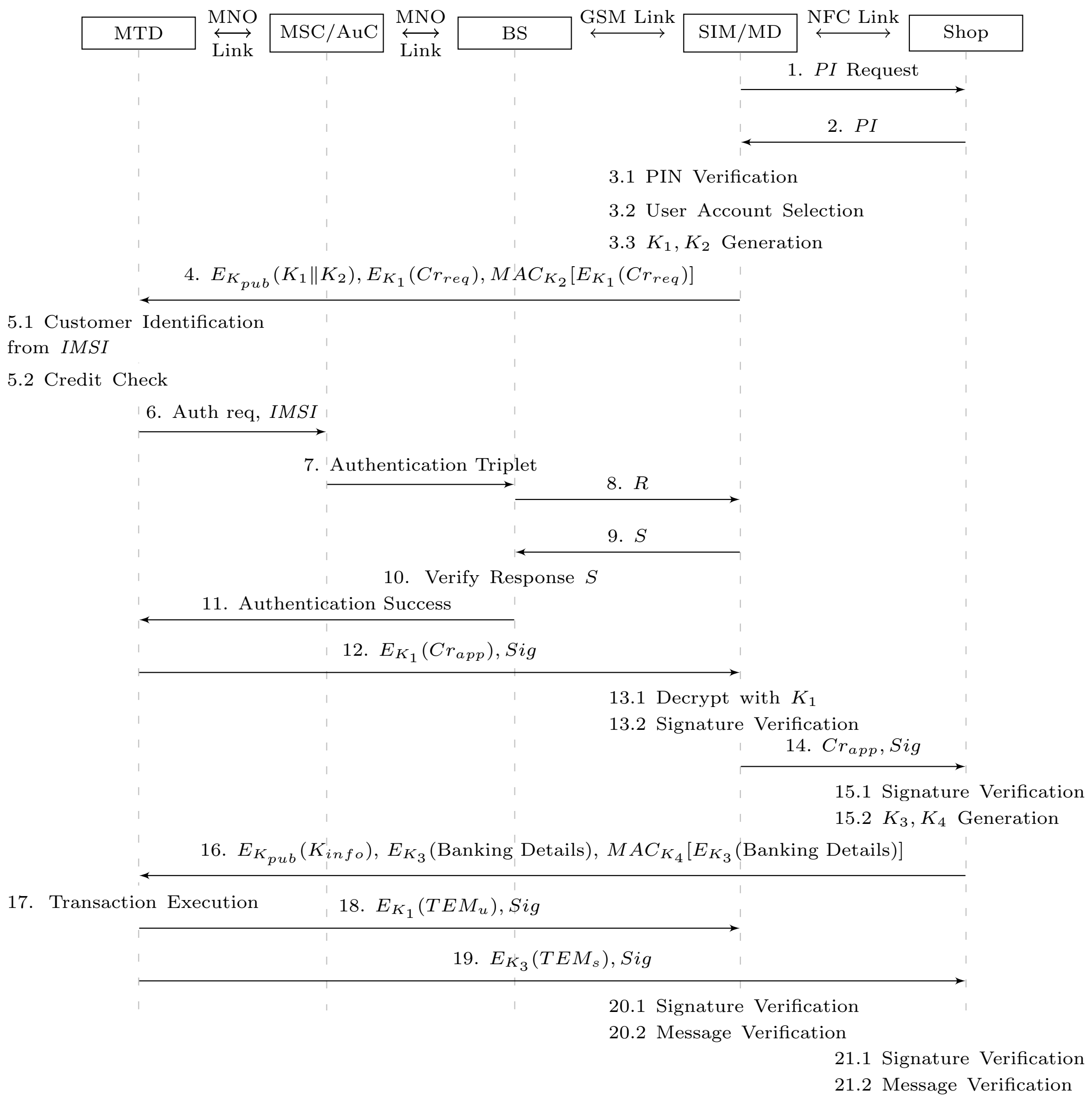
MTD
MNO Link
MSC/AuC
MNO Link
BS
GSM Link
SIM/MD
NFC Link
Shop
1. PI Request
2. PI
3.1 PIN Verification
3.2 User Account Selection
3.3 K_1, K_2 Generation
4. $E_{K_{pub}}(K_1 \| K_2), E_{K_1}(Cr_{req}), MAC_{K_2}[E_{K_1}(Cr_{req})]$
5.1 Customer Identification from $IMSI$
5.2 Credit Check
6. Auth req, $IMSI$
7. Authentication Triplet
8. R
9. S
10. Verify Response S
11. Authentication Success
12. $E_{K_1}(Cr_{app}), Sig$
13.1 Decrypt with K_1
13.2 Signature Verification
14. Cr_{app}, Sig
15.1 Signature Verification
15.2 K_3, K_4 Generation
16. $E_{K_{pub}}(K_{info})$, E_{K_3}(Banking Details), $MAC_{K_4}[E_{K_3}$(Banking Details)]
17. Transaction Execution
18. $E_{K_1}(TEM_u), Sig$
19. $E_{K_3}(TEM_s), Sig$
20.1 Signature Verification
20.2 Message Verification
21.1 Signature Verification
21.2 Message Verification

Figure 6.3: The Proposed Customer Authentication & Payment Protocol

6.4.1 Phase I: Customer Identification and Credit Check

This phase is initiated when the store owner sends the payment request to his NFC reader and the customer places his MD on the shop's NFC enabled point.

Step 1: The MD and shop terminal establish an NFC connection.

Step 2: The shop terminal forms the Payment Information message PI containing at least the Total Price TP, a temporary shop identity T_{SID}, and the shop's Time Stamp TS_s, and sends it to the MD:

$$PI = TP \| T_{SID} \| TS_s \tag{6.1}$$

The T_{SID} acts as one time identifier used by the shop to identify the transaction. It is updated and fresh for each transaction. Optionally, PI may also contain a description of the shop and the goods which would appear on the customer's credit/debit card account statement.

Steps 3-4: Once the payment information is received from the shop, the application installed on the MD displays the transaction amount TP to the user and asks him to select a payment account and provide PIN authentication. This is for assurance that the customer is the legal owner of the mobile device, and therefore also the owner of the account which will be used for payment. It also provides confirmation that the amount and account details are accepted by the user.

After successful PIN verification, the MD needs to obtain a credit approval certificate for the shop from the respective MTD indicating that the customer has sufficient funds in his account and has agreed to pay the required amount. The information in this exchange should not be accessible to the BS or any other entity of GSM network other than the MTD. To provide a secure connection for this exchange between the MD and the MTD, the mobile device generates two keys K_1, K_2 for symmetric encryption and MAC respectively. The actual encryption process used here is irrelevant, but will most likely be specified by the card provider and EMV requirements. It should not depend only on quantities known to either the BS or the MNO since only the MTD should be able to perform the decryption. The mobile device forms a credit request message Cr_{req} for credit approval from the MTD, namely,

$$Cr_{req} = PI \| IMSI \| Acc_{ID} \tag{6.2}$$

This is encrypted with K_1 and a MAC is computed on the ciphertext using K_2 to provide data integrity. Then the keys, K_1 and K_2, are encrypted with the MTD's public key K_{pub}. The entire message, consisting of the encrypted keys, the encrypted credit request and the MAC value, is sent to the MTD as in message 4 of Figure 6.3.

Step 5: Upon receipt of this message, the MTD starts by decrypting the first part of the message with its private key K_{pr} to extract the encryption and MAC keys, K_1 and K_2. It then verifies the MAC and in case of successful verification, it decrypts the second part of the message, containing Cr_{req}, and checks the freshness of the shop's time stamp in PI. The MTD identifies the customer from the IMSI in Cr_{req} and performs a credit check against the named account Acc_{ID}.

6.4.2 Phase II: Customer Authentication

Steps 6-11: Whether or not the credit and freshness checks are successful, the MTD sends an authentication request message to the MSC/AuC to authenticate the MD. The MD has already been identified by its IMSI. However, since the IMSI is not a secret, it may be used by a malicious party. To counter such threat, the MD needs to be authenticated under the IMSI claimed in Cr_{req} prior to any monetary transaction. With this IMSI, the MSC follows the usual procedure to authenticate an MD and it does not required further user interaction. So, in the case of successful authentication, the usual success message is sent from the BS to the MTD.

Step 12: If the credit check fails or the authentication success message is not received, the protocol is terminated with the sending of a fail message from the MTD to the MD. Termination does not occur before the authentication in order to hide the result of the credit check from an unauthenticated attacker. Otherwise, when both the credit check and the authentication are successful, a credit approval identifier App_{ID} is generated by the MTD. This acts as an index to a table in which the MTD stores information about the debit account, the amount to be transferred, the destination shop identity, a time stamp and the MD identity (IMSI). This identifier helps in resolving any disputes in the future but the details of the transaction are not contained therein.

The MTD now forms a new string Cr_{app} indicating credit approval for the Payment Information PI, namely,

$$Cr_{app} = PI \| TS_a \| App_{ID} \tag{6.3}$$

where TS_a is the MTD's approval time stamp.

The MTD computes a signature with its signing key K_{sign} over the hashed plaintext and encrypts the string Cr_{app} with the key K_1. The encrypted Cr_{app} along with its signature is transmitted to the mobile device. The former cannot be decrypted in transit as the encryption key K_1 is unavailable, nor is Cr_{app} revealed by applying the verification key to the signature because of the hashing. Moreover, because of TS_s, TS_a or App_{ID}, the message differs each time even if the user buys the same goods on successive occasions.

Steps 13-16: The mobile device decrypts the message with the encryption key K_1 to obtain Cr_{app} and forwards it to the shop along with the corresponding signature. The shop verifies the signature using K_{ver} and compares the PI content in the Cr_{app} message to the one it initially sent in message 2. In the case of an invalid signature or a mis-match with PI, the shop discards the message, rejects the payment, and withholds the goods or services from the customer. A successful verification indicates that the customer is legitimate and that the MTD has obtained agreement from the customer to pay. This is like a three party contract where a middle party (the MTD), trusted by both other parties, provides assurance that the other party is willing to pay the specified price.

The shop now needs to send its banking details to the MTD to complete the transaction. The banking details may include the account name and number, the bank and branch codes, etc. This is sensitive information and should not be disclosed to any entity other than the MTD, not even to the MD. The shop therefore generates encryption and MAC keys, K_3 and K_4 to secure its banking details. It encrypts the banking details with the key K_3, and computes a MAC over the ciphertext with the key K_4. It also forms a string, K_{info}, containing the information about the keys as follows:

$$K_{info} = K_3 \| K_4 \| App_{ID} \tag{6.4}$$

The role of the approval identifier App_{ID} in this step is to enable the MTD to connect the authentication phase to the transaction execution phase. The shop encrypts the string K_{info} with the public key K_{pub} of the MTD and sends it to the MTD via the MD. This forms a virtual tunnel between the shop and the MTD through the MD, as the latter cannot decrypt the message content. Note, however, that the shop needs to be certain it has K_{pub} correctly from the MTD, and not a key substituted by an attacker.

6.4.3 Phase III: Transaction Execution

Step 17: The MTD associates the App_{ID} received in the step 16 with the already stored App_{ID} (step 12). It decrypts the banking details of the shop with keys K_3, K_4 and transfers the approved amount, stored against corresponding App_{ID}, to the shop account. The MTD flags the App_{ID} indicating that the transaction has been executed to ensure that the same App_{ID} could not be used again.

Step 18-21 After a successful transaction, the MTD generates a Transaction Serial Number (TSN) and forms Transaction Execution Messages, TEM_u and TEM_s for the MD and the shop respectively.

$$\begin{aligned} TEM_u &= PI\|TSN\|TS_{tr}\|Acc_{ID} \\ TEM_s &= PI\|TSN\|TS_{tr}\|SBAD \end{aligned} \tag{6.5}$$

where,
TSN = Transaction serial Number
TS_{tr} = Time Stamp (transaction)
$SBAD$ = Shop Bank Account Details

The MTD computes a signature on the hashed plaintext, encrypts TEM_u with the key K_1, and sends it to the MD. The MD decrypts the message and verifies the signature. An invalid signature indicates that the transaction confirmation has been accidentally or deliberately corrupted en route. In such a case, the MD enquires about the transaction from the MTD. If the transaction has already been executed, the MD asks for a fresh confirmation message. Otherwise, it is obvious that message 16 has not been delivered to the MTD. This may happen if a malicious party has blocked the message from reaching the MTD and has instead transmitted a fake transaction confirmation message. Of course, such a fabricated message cannot go undetected as it is signed by the MTD. In such scenario, the MD asks the shop to resend message 16.

The MTD also forms a Transaction Execution Message, TEM_s, for the shop by appending the Shop's Banking Details as shown in Eq (6.5). The MTD computes a signature over the hashed plaintext and encrypts TEM_s with the key K_3. The MTD sends this encrypted message along with its signature to the customer MD which relays it to the shop. The customer's MD can neither decrypt this message as it does not possess K_3, nor alter any contents as they are protected by the signature. The shop decrypts the message, verify its contents and the signature, thereby confirming that his account (rather than an attacker's) has been credited correctly. The contents consist of important transaction information exchanged during the transaction. Hence, if the shop wants any subsequent clarification, it can approach the MNO quoting the TSN and the App_{ID} received in step 14. Finally, if the shop is satisfied, it produces a receipt together with the goods or services for the customer.

6.5 Analysis

In this section, we analyse the protocol from multiple perspectives to ascertain the strength of our protocol. This analysis encompasses the authentication and security of the messages. We assume that the MNO is trustworthy, whereas the customer or the shop can be dishonest, and there may be an active attacker listening to any of the NFC or other messages.

6.5.1 Dishonest Customer

Scenario 1. A dishonest customer plans to buy some products, making the payment from someone else's account. The PIN requirement in step 3 should force the customer to use his own mobile device to enact the protocol. Indeed the protocol depends on the strength of this PIN, just as is the case with credit card withdrawals. However, rogue applications on the MD could have already sniffed the PIN.

Assume that the attacker uses his own mobile and knows the IMSI and account numbers ($IMSI'$, Acc'_{ID}) of the target victim. He must fabricate Eq (6.2) as:

$$Cr'_{req} = PI \| IMSI' \| Acc'_{ID} \tag{6.6}$$

As this message can be decrypted only by the MTD, the malicious contents remain undetected by all other entities. The MTD decrypts the message and identifies the customer from $IMSI'$. Assuming the protocol does not fail here because the target victim is not a legitimate customer or the account has insufficient funds, the MTD proceeds to the fresh authentication of $IMSI'$. So the MSC/AuC provides the authentication triplet in step 7 corresponding to $IMSI'$. However, the attacker cannot compute the valid response S' as his mobile device lacks the necessary key K'_i. So, the authentication check fails and the protocol terminates. Thus, an incorrect identity cannot be successfully used in the protocol.

Scenario 2. Suppose a dishonest customer plans to buy goods without payment. He could accomplish this by providing his own banking details, instead of the shop's, to the MTD for the payment recipient. He then blocks the legitimate message 16, and replaces it as follows. Using his own keys K'_3 and K'_4, he fabricates message 16 with own banking details and sends it to the MTD. The MTD performs the transaction against this information, deducting the amount from the customer's account but paying it back into the same or another account of the customer. (These may be distinct in an attempt to avoid detection). After executing the transaction, the MTD sends 'receipts' in messages 18 and 19. The MD must block message 19 as this message contains the substituted bank details which the shop checks. So the dishonest customer needs to replace the banking details in this message with the shop's banking details. He can decrypt message 19 as it is encrypted with his own malicious key K'_3. However, he must now change the banking details and encrypt them with the shop's key K_3. As he lacks this key, he cannot generate a valid ciphertext. Moreover, the original message is protected by the digital signature. If the customer were to make any alteration to the banking details, it would void the signature which the shop verifies next. In neither case is the shop able to verify the transaction, and a failure message is reported to the shopkeeper. Hence, the dishonest customer is again unsuccessful.

There may be another approach to accomplish the above attack where the dishonest customer plans to buy some goods without payment. The dishonest customer does not communicate with the MTD since he could not succeed in the way described above; rather, he masquerades as the MTD to the shop. The target of the customer is to send fake but acceptable receipts to the shop at the end of the protocol by replaying old legitimate, messages or fabricating new messages. Since the customer is not communicating with the MTD, his account will not be debited. In the original protocol, the shop receives three messages from the MD: messages 1, 14 and 19. Message 1 originates from the MD, whereas messages 14 and 19 actually originate from the MTD but are relayed by the MD to the shop. The dishonest customer needs to construct or replay the latter two messages in such a way that they are acceptable to the shop. Both messages are digitally signed by the MTD. They contain the Shop Identifier T_{SID} and Time Stamp TS_s. T_{SID} is a random value generated by the shop every time at the start of the protocol. This value not only serves as a shop identifier during the protocol, but it also adds freshness to the protocol messages. TS_s is updated too in every protocol round, but it may be predictable to some extent. A combination of these two values, along with the digital signatures of the MTD, does not allow either replay or alteration of the messages to succeed. Hence, the dishonest customer is again unsuccessful. Of course, as usual in PKI, the shop should check the digital certificates of the MTD keys to justify its trust in them.

Scenario 3. Assume now that the dishonest customer plans to pay less than the required amount but claim payment of the full amount. To accomplish this, the MD sends TP' in the Credit Request message Cr_{req} of step 4 to the MTD, where $TP' < TP$. The MD receives the Credit Approval message, Cr_{app}, in step 12 from the MTD confirming that the initially requested amount TP' has been approved for transaction. But the MD needs to confirm to the shop in step 14 that the original amount, TP, is approved for transaction. Since the approved price is digitally signed by the MTD, it cannot be amended by the MD. So the actual price that is approved by the MTD is transmitted to the shop. As the shop application checks the approved amount against that requested, this attack also fails.

Scenario 4. Here, a dishonest customer wants to pay through a mobile device which he does not own. He might have stolen that device or found it as lost property. If the SIM is still valid and the credit/debit cards have not been cancelled, it can still be used for transactions. After the device receives the payment information PI from the shop in step 2, the application installed on the mobile device requires PIN verification from the customer. Since the customer does not own the mobile device, he should not have knowledge of the PIN. So the protocol does not proceed further. Additionally, the application can be designed to be blocked in the MD and by the MTD after a limited number of failed attempts at PIN verification. This provides an assurance to the customers that their lost mobile device could not be used for any monetary transactions even while the SIM remains active.

6.5.2 A Dishonest Shop

Scenario 5. The shop is dishonest and plans to draw more than the required amount without intimation to the customer. The information about the amount to be transferred is sent to the MTD by the MD in the Credit Request message, Cr_{req}, in step 4. A mobile device cannot send more than the price contained in PI and approved by the user in step 3 unless the device itself is compromised. Therefore, a shop cannot obtain more than the agreed amount if, as requested, the customer checks the amount before entering his PIN.

Scenario 6. The shop is dishonest and denies receipt of the transaction execution message in step 19. In this way, the shop decides not to deliver the goods or services despite receiving the required amount. However, the MD has the signed receipt from the MTD with the TSN from Eq (6.5). This is linked to the approval App_{ID} generated in step 12. As both are digitally signed by the MTD, the customer can approach the MTD regarding any dispute. With knowledge of the account credited during the transaction and the shop receipt from the customer, the MTD can take action to identify the criminal and refund the customer.

6.5.3 Message Security

Apart from the above-mentioned scenarios, we also analysed our protocols from various other angles. The data over the GSM link (between the MD and the BS) is encrypted according to the GSM specification. The data sent over the NFC link in steps 1, 2 and 14 are sent in the clear. This data does not contain any particularly sensitive information except perhaps for the TP. However, the range within which this data can be captured is very limited, and it is occupied by the shop keeper and the customer, at least one of whom should notice unwelcome devices (such as other NFC capable mobile phones) in the vicinity. The read range of the price displayed on both the shop till and the user's MD is much more than the range of the NFC link. Therefore, we considered PI as not sufficiently sensitive to need protection over the NFC link. Nevertheless, we should consider this in a little more detail.

Other information that is sent in clear over the NFC link includes the App_{ID} in the Cr_{app} message. At this point the attacker can hi-jack the protocol by blocking the communication of message 16, replacing it with his own forged message which contains his own bank details. There is no relevant data which is not known to the attacker. This results in a successful transfer of funds to the criminal and also a successful acknowledgment in step 18 to the legitimate customer. However, the shop owner will either not receive the transfer message in step 20, or will receive one which fails his verification. Thus, although the shop keeper will not then release the goods, the attacker will have obtained the funds. The solution is to include a means for the MTD to verify that message 18 comes from the same source as message 2.

We therefore propose the inclusion of a Diffie-Hellman key agreement (DH) between the MD and Shop during messages 1 & 2 in situations where the NFC link may be compromised. Then step 18 can include a proof of origin. Step 1 would include the public parameters for DH, and the MD's exponentiated value, while step 2 would include the shop's response of the other DH exponentiated value. As message 16 contains a MAC of the other components of message 16 using the DH shared key, the MD can check the authenticity of message 16, ensuring that the protocol has not been hi-jacked. However, an attacker who can hi-jack the protocol at step 18 could equally easily hi-jack it at step 1. This requires blocking the legitimate message PI and replacing it as necessary with PI' so that the MD agrees a shared key with the attacker instead of the shop and, later, the forged message 16 is authenticated by the MD. Since TP is not known to the attacker until PI is transmitted, the attacker needs to collect the legitimate PI first in order to include TP in PI', this being necessary to obtain the customer's agreement over the price. However, for this to succeed, the attacker must prevent the correct PI from reaching the MD. Consequently, the success of Diffie-Hellman key exchange between shop and MD cannot be prevented unless the attacker can guess TP correctly or the customer fails to check the amount carefully. An attacker may use a hidden camera which can read the shop's till display, then his NFC hi-jack device can know TP in advance and so determine a value for PI which the customer will accept. He can therefore block the legitimate message 2 and replace it with his own. The threat from this is similar to that of a camera capturing PIN values. Payment Card Industry Security Standard Council (PCI SSC) prohibits use of cameras near a PIN entering device to avoid monitoring of displays, PIN pads, etc., [8]. Moreover, there are two methods to block RF communication on the NFC link and neither of the methods can easily be adopted in our scenario. The first is to cover the transmitter or receiver with some shielding material. The other method is to produce a high noise on the same operating frequency resulting in a significant decrease in the signal-to-noise ratio. For the former the attacker must shield the MD or the shop terminal. The latter requires noise generating hardware in close proximity to the MD and the shop sales terminal. Both approaches are visibly detectable. This means there is little scope for a successful attack when the MD also verifies the authenticity of message 16. It should therefore be an acceptably small risk.

App_{ID}, which is sent in the clear over the NFC link, is a random string generated by the credit approval authority. From an attacker's perspective, its only significance is its assurance that the customer had, at least before the transaction, the amount TP in his account. This assurance can also be achieved if a customer successfully pays for some goods. Therefore, App_{ID} is not sensitive information in this scenario.

The Role of the Approval Identifier in message 16. App_{ID} acts as a bridge between phase II and III. It also adds freshness to message 16, so it cannot be replayed in future. Any alteration in the K_{info} results in invalid keys and an invalid App_{ID}. Hence it is detectable.

Non-repudiation of Transaction Execution Messages. TEM_u and TEM_s are digitally signed by the MTD. In case of any dispute over payment, the MTD has to honour both messages. So, both the customer and the shop are completely assured of the transaction payment taking place.

Disclosure of Relevant Information. The Cr_{req} containing price information is not disclosed to the base station or any other GSM entity apart from the MTD. The SBAD is sensitive information. It is encrypted not only over the GSM links but also over the NFC link. It is transmitted through the mobile device to the MTD, yet the former cannot decrypt this information. The Acc_{ID} of the customer is not disclosed to the shop. The MNO does not need to know the shopping details of the customer. Therefore, only the total amount is communicated to the MNO for transaction.

New Keys for every Transaction. The encryption and MAC keys for the message Cr_{req}, namely K_1 and K_2, are freshly generated by the mobile device in each round. Similarly, the keys K_3 and K_4, generated by the shop, are fresh for each transaction. Of course, these should not be predictable, especially if previous such keys become known.

Encryption and MAC Keys. Separate keys are used for encryption and MAC calculation making the protocol more secure. *Encrypt-then-MAC* is an approach where the ciphertext is generated by encrypting the plaintext and then appending a MAC of the encrypted plaintext. This approach is cryptographically more secure than other approaches [15]. Apart from its cryptographic value, the MAC can be verified without performing decryption. So, if the MAC is invalid for a message, the message is discarded without decryption. This results in computational efficiency.

6.5.4 Monetary Transaction Between Two Individuals

The proposed protocol can be used for monetary transactions between two individuals. The payee acts as a shop PoS terminal, and can use his own mobile phone for this. The added advantage in our proposal is that the payee does not need to register himself with the payer's (= customer's) MNO to receive a payment. This eliminates dependency of both parties to be on the same mobile network for monetary transactions. The payee needs only to provide his banking details in step 16 of the protocol.

6.5.5 Comparison of Proposed and Conventional Framework

Our proposed model is a Card not Present (CNP) model. If we compare our model with a conventional payment system described in 6.2.2, it is obvious that the MTD plays the role of an acquirer in our framework. The main difference is in the authentication of the user, which in our case is carried out through GSM network. In conventional CNP models, only the visible card data such as card number, cardholder's name, expiry date are used to authenticate the user. CNP transactions are vulnerable to fraud as the chip data cannot be verified [18]. In our model, we virtually linked a registered bank card to a SIM card against a specific user. Now, the SIM card is also authenticated along with the CNP verification. Like an acquirer, the MTD acts a third party between the merchant and the card holder for monetary transfer. It also keeps a record of all transactions for any future disputes.

The issuer in our model, for *pay-as-you-go* transactions, is the same as the issuer in the conventional payment system, i.e., the financial entity that issues card to a card holder. The MTD communicates with the issuer to debit the account of the card holder and then credit the account of the shop. On the other hand, for *pre-paid*, the issuer is subsumed by the MTD.

6.6 Conclusion

In this chapter, we have proposed another transaction protocol for providing a secure and trusted communication channel for payment of goods using mobile devices. We used a different approach from the earlier versions of this protocol. The main advantage of this protocol over its earlier version is the elimination of a shared secret between the shop and the MNO. This makes this protocol more flexible and can also be used for monetary transactions between two individuals even though both use different mobile networks. We proposed a dedicated department, MNO Transaction Department (MTD), responsible for all the transactions. The security features of our protocol provide virtual tunnels among the mobile device, shop PoS terminal and the MTD. This caters for unsecured channels within various entities of the GSM network. The analysis shows the protocol is secure against various attacks by a dishonest customer or a dishonest shop.

In the next chapter, we will shift our focus to light-weight (e.g., NFC Type-1 and Type-2 tags) RFID tags. We will describe a mutual authentication protocol for light-weight RFID tags with some attacks to recover the secret keys.

Chapter 7

Attacks on Authentication Protocols

This is a joint work with Zeeshan Bilal and Keith Martin (ISG, RHUL). My contribution in this work is the Full Disclosure Active(FDA) Attack on SIDRFID (§ 7.4.2). Moreover, I also contributed in many fruitful discussions that resulted in various other attacks mentioned in the chapter. This work is published in the Journal of Applied Mathematics and Information Sciences [17]. This chapter is almost verbatim of the published paper.

7.1 Introduction

We have, till now, covered the authentication issues related to NFC technology. Our proposals for tag authentications (described in Chapter 4) deals with only high-cost NFC tags that are able to perform public key cryptography. The light-weight NFC tags, such as Type-1 or Type-2 tags described in §2.2 are unable to perform public key cryptography. These tags require light-weight easily implementable authentication mechanisms. NFC, being an extension to RFID technology, can incorporate RFID authentication protocols. We selected an ultra-lightweight mutual authentication protocol for RFIDs proposed by Yung-Cheng Lee [43] that can also be used in NFC tags and performed its security analysis. We discovered that the Lee's protocol is vulnerable to multiple attacks described §7.4 and §7.5 of this chapter.

7.2 Authentication in RFIDs

Radio Frequency Identification (RFID) systems are becoming pervasive in large scale identification applications [39]. The most widely deployed are low-cost RFID systems [23], where tags normally cost a few cents. These tags are likely to replace bar-codes as the line of sight is not required making it more user-friendly. However, there are many privacy and security concerns with low-cost RFID systems [39]. The main limiting factor in low-cost RFID tags relates to lack of resources, such as memory, computational power etc. RFID tags can be *roughly* classified into four classes based on the available resources [23].

1. The *full-fledged* class refers to those RFID tags that have enough resources to support conventional cryptographic functions like symmetric encryption, cryptographic one-way function, or even the public key algorithms. NFC Forum Type 4 tags or the tags used in ePassport fall in this category.

2. The *simple* tags refers to those tags that have sufficient power to support simple functions such as random number generator or one-way hashing function.

3. The third class called *lightweight* refers to those tags that can support simple functions like Cyclic Redundancy Code (CRC) checksum, or a random number generator but not a hash function.

4. The fourth class is *ultralightweight* referring to the tags that only support simple bitwise operations (like XOR, AND, OR, etc).

Consequently, authentication schemes used in RFID tags also follow similar categorization. We focus on Low-cost RFID systems that fall in the ultra-lightweight class of RFID tags. Yung-Cheng Lee proposed two ultra-lightweight authentication protocols, *SIDRFID* and *DIDRFID*, for RFID tags [43]. In the *SIDRFID*, the tags and the reader do not share any secrets, but rather use their respective identities as shared secrets. These identities are, therefore, sensitive information so they are not transmitted in clear. These identities are not updated and are static. Hence the protocol is termed as "Ultra-lightweight RFID Protocol with Static Identity *(SIDRFID)*". In the *DIDRFID*, the tag and the reader share a secret key K. The K and the tag identity, IDT are updated in each authentication session. Therefore this protocol is called "Ultra-lightweight RFID Protocol with Dynamic Identity *(DIDRFID)*". Both protocols claim to provide mutual authentication and implement very efficient and extremely lightweight functions. We discuss these protocols in greater depth in Section 7.3.

Avoine *et al.* have carried out a security analysis of both protocols [14]. They observe that using a single master key in *SIDRFID* is a single point of failure if compromised. However, they do not elaborate on any specific technique to recover the master key. We show in this paper how to recover this single master key and break the entire *SIDRFID* system. Further, Avoine *et al.* highlight an attack on the secret key used in *DIDRFID*. This attack involves eavesdropping two rounds of the authentication session and L^2 possible guesses (where L is the length of secret key K). We improve this attack and demonstrate a passive full disclosure attack on the *DIDRFID*. Our attack determines the correct key after eavesdropping approximately $\sqrt{\pi L}$ authentication sessions.

Our analysis is explained in detail in Section 7.4 and Section 7.5. We also describe further attacks on these protocols including one where an attacker successfully traces a tag.

7.3 Two Ultra-lightweight Authentication Protocols

In this section, we summarize the two authentication protocols proposed by Yung-Cheng Lee for low-cost RFID systems [43]. These protocols belong to the ultra-lightweight class and claim to provide mutual authentication. Additionally, these protocols claim to resist attacks including traceability, replay, de-synchronization and impersonation. Importantly, the computation cost is kept low by incorporating lightweight functions. In the proposed protocols, the pseudo-random number generator is installed only in the reader. The low-cost tag only performs simple bit-wise operations (XOR, AND, OR) and bits rotation $Rot(A, B)$, where $Rot(A, B)$ represents left rotation of string A by $HW(B)$ bits.

7.3.1 Static Identity Protocol for RFID *(SIDRFID)*

The protocol assumes that tag and reader each have static identities IDT and IDR, respectively, which are secret values shared by each entity (it is assumed that tag and reader have these pre-installed prior to activation of the scheme). The IDR is also stored in the tag which implies that this protocol can only be implemented in scenarios where there is one particular reader or many readers with the same IDR. The protocol executes as follows:

- **Step 1.**
  - Reader generates R

- Reader computes:

$$S_i = R \oplus IDR$$

- $Reader \rightarrow Tag : S$

- **Step 2.**
  - Tag computes:

$$R = S \oplus IDR$$
$$P = IDT \oplus Rot(R, IDR)$$
$$Q = Rot(IDT, IDT) \oplus Rot(R, R)$$

  - $Tag \rightarrow Reader : (P, Q)$

- **Step 3.**
  - Reader computes:

$$IDT = P \oplus Rot(R, IDR)$$
$$Q^{'} = Rot(IDT, IDT) \oplus Rot(R, R))$$

  - Reader authenticates tag as follows:

```
if $Q^{'} = Q$ then
    Tag is authenticated
else
    Protocol is abandoned
end if
```

- **Step 4.**
  - In case of successful tag authentication, the reader computes:

$$Z = Rot(IDT, IDR \oplus R) \oplus Rot(IDR, IDT \oplus R)$$

  - $Reader \rightarrow Tag : Z$

- **Step 5.**

- Tag computes:

$$Z^{'} = Rot(IDT, IDR \oplus R) \oplus Rot(IDR, IDT \oplus R)$$

- Tag authenticates reader as follows:

```
if $Z^{'} = Z$ then
    Reader is authenticated
else
    Protocol is abandoned
end if
```

7.3.2 Dynamic Identity Protocol for RFID (*DIDRFID*)

The protocol assumes that tag and reader share a secret key K . This may be pre-installed in the reader and in the tag prior to activation of the scheme. The dynamic ID of the tag, $DIDT$, and the secret key K are updated after every authentication session. We use $DIDT_i$ and K_i as the dynamic ID of the tag and the secret key respectively in the i^{th} authentication session. The protocol, in the i^{th} session, executes as follows:

- **Step 1.**
  - $Tag \rightarrow Reader : DIDT_i$

- **Step 2.**
  - Reader uses $DIDT_i$ as index to extract the corresponding secret key K_i from the database.
  - Reader generates a random number R_i.
  - Reader computes:

$$A_i = K_i \oplus R_i$$
$$B_i = Rot(K_i, K_i) \oplus Rot(R_i, R_i)$$

  - $Reader \rightarrow Tag : (A_i, B_i)$

- **Step 3.**

- Tag computes:

$$R_i = A_i \oplus K_i$$
$$B_i^{'} = Rot(K_i, K_i) \oplus Rot(R_i, R_i)$$

- Tag authenticates reader as follows:

```
if $B_i^{'} = B_i$ then
    Reader is authenticated
else
    Protocol is abandoned
end if
```

- **Step 4.**

  - In case of successful reader authentication, the tag computes:

$$C_i = Rot(K_i, R_i) \oplus Rot(R_i, K_i)$$

  - $Tag \rightarrow Reader : C_i$

- **Step 5.**

  - Reader computes:

$$C_i^{'} = Rot(K_i, R_i) \oplus Rot(R_i, K_i)$$

  - Reader authenticates tag as follows:

```
if $C_i^{'} = C_i$ then
    Tag is authenticated
else
    Protocol is abandoned
end if
```

- **Key Updating Step.** After successful mutual authentication, tag and reader update their values:

  - Tag and Reader compute:

$$DIDT_{i+1} = Rot(R_i, R_i \vee K_i) \oplus Rot(K_i, R_i \wedge K_i)$$
$$K_{i+1} = Rot(R_i, R_i \wedge K_i) \oplus Rot(K_i, R_i \vee K_i)$$

– Tag and Reader both keep $(DIDT_i, K_i)$ and $(DIDT_{i+1}, K_{i+1})$ in their memory.

7.4 Security Analysis of *SIDRFID*

In this section, we carry out a security analysis of *SIDRFID* [43]. Avoine *et al.* have suggested that *SIDRFID* is a weak protocol because it uses a single master key which in many situations is considered unacceptable [14]. However, there may be applications, such as issuing temporary RFID tags for access control to a team visiting an organization, where use of a single master key may be justified. In such scenarios, we do not need to generate new keys on every access attempt and thus avoid the need for secure distribution of these secret keys to each tag. Nonetheless we show that, even in situations where a fixed master key is justified, the secret entities can be easily recovered thus demonstrating that *SIDRFID* is a very weak protocol.

7.4.1 Passive Hamming Weight Disclosure (*PHWD*) Attack

We first present a passive attack which reveals HW(IDR). We make the realistic assumption that the channel between the tag and the reader is wireless and insecure. The attacker simply needs to eavesdrop any two authentication sessions. Moreover, the resources available to the attacker are also limited so it cannot perform heavy computations (a realistic assumption in lightweight cryptography). The attack executes as follows:

- **Step 1.** Attacker eavesdrops two legitimate authentication sessions to obtain S_1, P_1 and S_2, P_2.

- **Step 2.** The attacker computes A and B as follows:

$$\begin{aligned} A &= S_1 \oplus S_2 \\ &= (R_1 \oplus IDR) \oplus (R_2 \oplus IDR) \\ &= R_1 \oplus R_2 \end{aligned} \tag{7.1}$$

$$\begin{aligned} B &= P_1 \oplus P_2 \\ &= (IDT \oplus Rot(R_1, IDR)) \oplus (IDT \oplus Rot(R_2, IDR)) \\ &= Rot(R_1, IDR) \oplus Rot(R_2, IDR) \\ &= Rot(R_1 \oplus R_2, IDR) \end{aligned} \tag{7.2}$$

From (7.1) and (7.2), we get:

$$B = Rot(A, IDR) \tag{7.3}$$

Since A and B are known from (7.1) and (7.2), $HW(IDR)$ can easily be obtained from (7.3).

After disclosing $HW(IDR)$, an attacker can carry out a selective brute force attack to find the exact value, where each value has correctness probability (considering L as the length of bit string IDR):

$$p = \frac{1}{\binom{L}{HW(IDR)}}$$

This value is much higher than 2^{-L}, which is the probability of brute force attack success against an L-bit value. If we assume IDR to be similar to those assigned as EPC values (96-bits [5]), then the IDR consists of only 36 unknown bits (which we denote IDR^*) and the remaining 60 bits are publicly known (these determine the header, manufacturer and type of item details). This further raises the correctness probability p' of a guess to:

$$p' = \frac{1}{\binom{36}{HW(IDR^*)}}$$

The denominator's maximum value reaches to 2^{33} corresponding to $HW(IDR^*) = 18$. As $HW(IDR^*)$ is already obtained from Eq (7.3), an attacker needs 2^{33} attempts, at maximum, to recover the key which is substantially fewer trials to conduct.

7.4.2 Full Disclosure Active (*FDA*) Attack

We now present a Full Disclosure Active (*FDA*) attack against *SIDRFID*. We assume that either the attacker is in possession of the tag or there is no restriction on accessing the tag. This attack involves eavesdropping one authentication session and sending $L - 1$ chosen public messages to the tag, where L is the length of bit string IDR. The *FDA* attack is explained as follows:

- **Step 1.** The attacker eavesdrops a legitimate authentication session and records S_1, P_1, Q_1 and Z_1 (described in Section 7.3.1), where the labels of individual bits in each of these strings is as for the string X in the List of Notation.

- **Step 2.** The attacker impersonates a legitimate reader and sends S_2, a manipulated version of S_1 with the two least significant bits flipped as $s_0^{'}$ and $s_1^{'}$ (the subscript of S represents the authentication session and subscript of s represents the bit position).

- **Step 3.** Tag computes R_2 as follows:

$$R_2 = S_2 \oplus IDR \tag{7.4}$$

Since IDR is fixed, R_2 is the same as R_1 except that the least significant two bits are flipped as $r_0^{'}$ and $r_1^{'}$ as follows:

$$\begin{aligned} R_1 &= r_{L-1} \cdots r_2 r_1 r_0 \\ R_2 &= r_{L-1} \cdots r_2 r_1^{'} r_0^{'} \\ \text{Let } M &= R_1 \oplus R_2 \\ &= 00 \cdots 011 \end{aligned} \tag{7.5}$$

Tag now computes P_2 and Q_2 where,

$$\begin{aligned} P_2 &= IDT \oplus Rot(R_2, IDR) \\ Q_2 &= Rot(IDT, IDT) \oplus Rot(R_2, R_2) \end{aligned}$$

and sends them to the attacker.

- **Step 4.** After receiving P_2 and Q_2, the attacker computes N as:

$$\begin{aligned} N &= P_1 \oplus P_2 \\ &= (IDT \oplus Rot(R_1, IDR)) \oplus (IDT \oplus Rot(R_2, IDR)) \\ &= Rot(R_1, IDR) \oplus Rot(R_2, IDR) \\ &= Rot(R_1 \oplus R_2, IDR) \\ N &= Rot(M, IDR) \end{aligned} \tag{7.6}$$

Since N and M are known in (7.6), HW(IDR) can be calculated.

- **Step 5.** The attacker now computes T as:

$$\begin{aligned} T &= Q_1 \oplus Q_2 \\ &= (Rot(IDT, IDT) \oplus Rot(R_1, R_1)) \oplus (Rot(IDT, IDT) \oplus Rot(R_2, R_2)) \\ &= Rot(R_1, R_1) \oplus Rot(R_2, R_2) \end{aligned} \tag{7.7}$$

- **Step 6.** R_2 is same as R_1 except that the least two bits are flipped as $r_0^{'}$ and $r_1^{'}$, as explained before for deriving (7.5). The two least significant bits of R_1, will either be the same or different with probability one half. The attacker thus analyses (7.7) according to two conditions as follows:

  1. **Case 1.** The two flipped bits of R_1 are different, which results in:

     $$HW(R_1) = HW(R_2)$$

     This simplifies (7.7) as follows:

     $$\begin{aligned} T &= Rot(R_1 \oplus R_2, R_1) \\ &= Rot(M, R_1) \end{aligned} \tag{7.8}$$

     Since M is a string of all 0's except for two consecutive 1's in the least significant positions (as described for (7.5)), T will also consist of all 0's except for two consecutive 1's in the string. The position of the first 1 starting with the least significant bit as zero determines HW(R_1). The attacker marks the least significant bit of R_1 as x and the next bit as $x^{'}$ (in this case the first two LSBs are inverses of each other).

  2. **Case 2.** The two flipped bits of R_1 are the same which results in either:

     $$HW(R_1) = HW(R_2) + 2$$

     or

     $$HW(R_1) = HW(R_2) - 2$$

     Since HW(R_1) $\neq$ HW(R_2), this does not simplify (7.7). In this case the string T will be a random string of 0's and 1's without any pattern. The attacker marks the least significant bit of R_1 as x and the next bit as x, since both bits are either 0 or 1.

- **Step 7.** The attacker continues sending the next chosen plaintext S_3 by flipping (s_0, s_2). The resultant string T in this case will reveal whether r_2 is the same as r_0.

  ```
  **if** $r_2 = r_0$ **then**
      $r_2 = x$
  **else**
      $r_2 = x^{'}$
  **end if**
  ```

  In general, the attacker continues sending chosen plaintexts by flipping two bits (s_0, s_k) where $k = 1 \cdots (L-1)$ as shown in Figure 7.1. For every k^{th} authentication session, the string T in (7.7) reveals two bits of R_1, (r_0, r_k), to be either the same or otherwise.

- **Step 8.** At the end of this attack, R_1 is represented as a string of x and $x^{'}$ with known HW(R_1) from (7.8). The attacker now replaces x's with 1's and $x^{'}$'s with 0's, or *vice versa* according to HW(R_1).

- **Step 9.** The only non-trivial value will be when HW$(R_1) = L/2$. In this case, x can either be 1 or 0. Thus, R_1 has two possible values. The attacker uses the eavesdropped legitimate round of Step 1 and checks which of the two possible values of R_1 satisfies the values of the public messages S_1, P_1 and Q_1.

- **Step 10.** Once we get the value of R_1, we can easily determine IDR and IDT from any of the public messages. It now becomes very easy to launch multiple attacks on a tag including tag cloning, tag tracking and inventorying [39].

7.4.3 Other Attacks

We have just shown a full disclosure attack which completely disrupts the authentication process in *SIDRFID*. We now highlight further weaknesses in the design of this protocol which can be exploited to launch multiple attacks.

Traceability Attack

We assume that a low-cost RFID tag is unable to keep track of the current status in an authentication round. It thus replies to every query sent by a compatible reader.

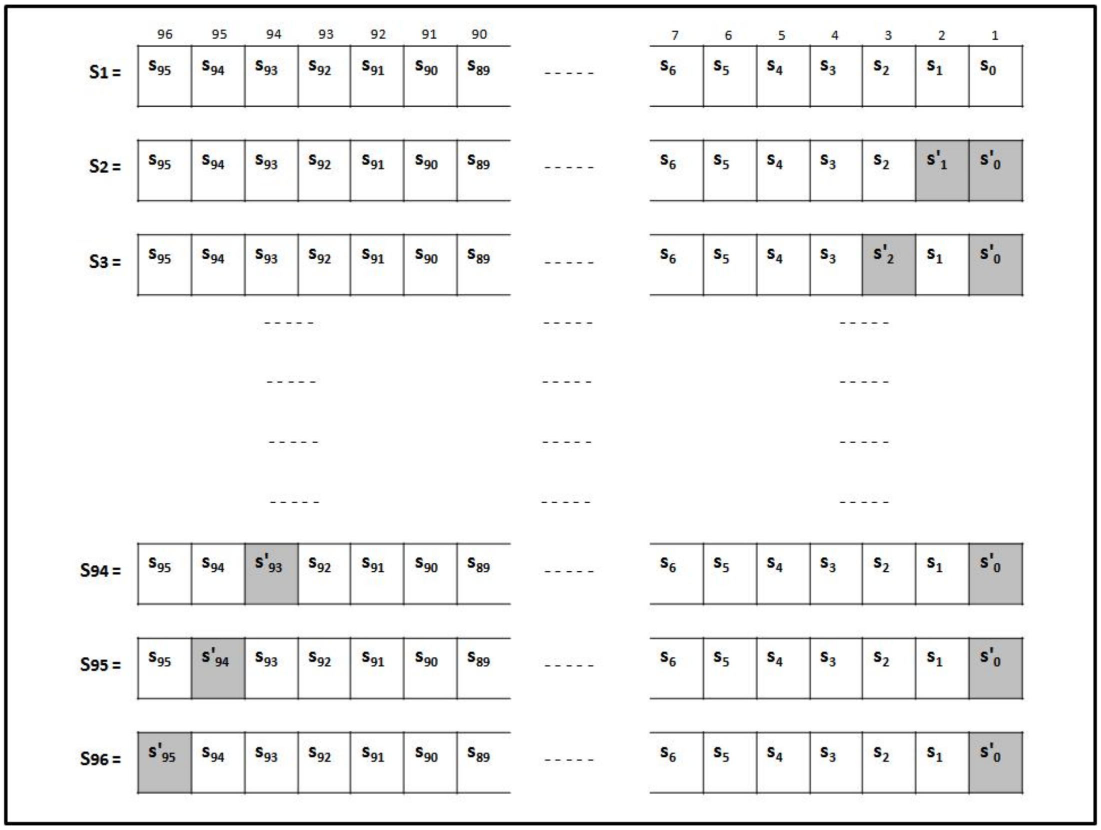
96 95 94 93 92 91 90 ... 7 6 5 4 3 2 1
S1 = s_{95} s_{94} s_{93} s_{92} s_{91} s_{90} s_{89} ----- s_6 s_5 s_4 s_3 s_2 s_1 s_0
S2 = s_{95} s_{94} s_{93} s_{92} s_{91} s_{90} s_{89} ----- s_6 s_5 s_4 s_3 s_2 s'_1 s'_0
S3 = s_{95} s_{94} s_{93} s_{92} s_{91} s_{90} s_{89} ----- s_6 s_5 s_4 s_3 s'_2 s_1 s'_0
S94 = s_{95} s_{94} s'_{93} s_{92} s_{91} s_{90} s_{89} ----- s_6 s_5 s_4 s_3 s_2 s_1 s'_0
S95 = s_{95} s'_{94} s_{93} s_{92} s_{91} s_{90} s_{89} ----- s_6 s_5 s_4 s_3 s_2 s_1 s'_0
S96 = s'_{95} s_{94} s_{93} s_{92} s_{91} s_{90} s_{89} ----- s_6 s_5 s_4 s_3 s_2 s_1 s'_0

Figure 7.1: Full Disclosure Active (*FDA*) Attack in *SIDRFID* (for *L*=96).

In *SIDRFID*, the public messages P and Q are different in every authentication session because of the different random R's generated by the reader. The attacker thus eavesdrops one round of authentication and keeps on sending the same S, thus forcing the tag to calculate similar public messages. This will facilitate tracking of a particular tag.

Reader Impersonation

The order of authentication is important in RFID authentication protocols and can counter several active attacks. The reader should be authenticated first so the tag may transmit its secret information only to a legitimate reader. The wrong order of authentication leads to a reader impersonation attack. An attacker can eavesdrop a legitimate authentication round. The attacker can then impersonate a legitimate reader and replay the eavesdropped response as legitimate and get itself authenticated. This attack is possible because secret values are not updated in each fresh round of authentication.

Identification of Reader

SIDRFID does not specify how the tag determines which IDR is to be used to generate the public values. Therefore, a further limitation of this protocol is that it can only be implemented in scenarios where there is only one particular reader or many readers with the same IDR.

7.5 Security Analysis of *DIDRFID*

In this section, we carry out a security analysis of *DIDRFID* [43]. Avoine *et al.* presented a key guessing attack against *DIDRFID* [14]. This attack requires eavesdropping two authentication session and a total of L^2 possible guesses, where L is the length of the secret key. Whilst this is a serious attack, we present another variant of full disclosure attack which uniquely determines the key in many fewer attempts. This further demonstrates that *DIDRFID* is a very weak protocol.

7.5.1 Passive Weight Disclosure (*PWD*) Attack

We assume that the channel between the tag and reader is wireless and insecure. The *PWD* attack first obtains $\mathrm{HW}(K)$ which then allows us to uniquely determine the correct secret K.

The details of this protocol are given in § 7.3.2 and our attack, which extracts the secret key K, is as follows:

- **Step 1.** Attacker scans the communication channel until he observes that the message B_i in (7.9) sent by reader to tag (forward channel) is same as the message C_i in (7.10) sent by tag to reader (backward channel).

$$B_i = Rot(K_i, K_i) \oplus Rot(R_i, R_i) \tag{7.9}$$

$$C_i = Rot(K_i, R_i) \oplus Rot(R_i, K_i) \tag{7.10}$$

It is evident from (7.9) and (7.10) that $B_i = C_i$ when:

$$HW(K_i) = HW(R_i) \tag{7.11}$$

The probability P of meeting this condition for two random L bits values is:

$$P = \frac{1}{(2^L)^2}\sum_{i=0}^{L}\binom{L}{i}^2 \tag{7.12}$$

We use an approximation of Eq 7.12 using Vandermonde convolution formula (also called ChuVandermonde formula) [32, 66] and Stirling's approximation [28]. Using Vandermonde convolution formula, we get:

$$\sum_{i=0}^{L}\binom{L}{i}^2 \approx \binom{2L}{L} \tag{7.13}$$

and the Stirling's approximation provides:

$$\binom{2L}{L} \approx \frac{4^L}{\sqrt{\pi L}} \tag{7.14}$$

Hence it follows that:

$$P \approx \frac{1}{\sqrt{\pi L}} \tag{7.15}$$

So, we say that the attacker needs to observe $\sqrt{\pi L}$ authentication sessions (on average) to find the message B sent by the reader equal to the message C sent by the tag. For $L = 96$ (assuming that the scheme is following EPC standard), the attacker eavesdrops only 18 sessions of authentication, on average, to get the required outcome.

- **Step 3.** Once the condition in Eq (7.11) is satisfied, attacker re-writes (7.9) and (7.10) as follows:

$$B_i = Rot(K_i \oplus R_i, K_i), \tag{7.16}$$

$$C_i = Rot(K_i \oplus R_i, K_i) \tag{7.17}$$

- **Step 4.** Since message A is:

$$A_i = K_i \oplus R_i. \tag{7.18}$$

as described in Section 7.3.2, (7.16) and (7.17) can be written as:

$$B_i = C_i = Rot(A_i, K_i). \tag{7.19}$$

Since A_i, B_i and C_i are known, HW(K_i) can be computed from (7.19) which will be the same as HW(R_i) according to (7.11).

- **Step 5.** Since A_i, HW(K_i) and HW(R_i) are known, the attacker uses (7.18) to determine j (the number of 1's in K_i overlapping with R_i at the same bit positions). The value of j is computed as follows:

$$j = HW(K_i) - \frac{1}{2}HW(A_i) \tag{7.20}$$

- **Step 6.** The value of j from (7.20) is used to determine $HW(R_i \vee K_i)$ and $HW(R_i \wedge K_i)$ as follows:

$$HW(R_i \vee K_i) = HW(A_i) + j \tag{7.21}$$

$$HW(R_i \wedge K_i) = j \tag{7.22}$$

- **Step 7.** The attacker now $XORs$ the update equations given in Section 7.3.2 as follows:

$$\begin{aligned} DIDT_{i+1} \oplus K_{i+1} &= Rot(R_i \oplus K_i, R_i \vee K_i) \oplus Rot(R_i \oplus K_i, R_i \wedge K_i) \\ &= Rot(A_i, R_i \wedge K_i) \oplus Rot(A_i, R_i \vee K_i) \end{aligned} \tag{7.23}$$

DID_{i+1} is transmitted in the next authentication session, where A_i, HW($R_i \vee K_i$) and $HW(R_i \wedge K_i)$ are already known. So K_{i+1} can be easily and uniquely calculated from the above equation.

7.5.2 Comparison between Our Attack and Avoine's Attack

The complexity of revealing the secret K for both attacks depends on the number of bits of secret K. The number of operations in Avoine's attack corresponds to the number of guesses before revealing the correct K. Avoine's attack thus requires a total of L^2 guesses and eavesdropping of two sessions of the *DIDRFID* protocol.

Our attack requires a small number of authentication sessions ($\sqrt{\pi L}$) as mentioned in Eq 7.15 to be eavesdropped, but once this is done there is no further "guesswork" required since the key K is uniquely revealed.

We note that for the case of EPCglobal tag, $L = 96$ and hence the attacker needs to eavesdrop 18 authentication sessions, on average. Since eavesdropping the tag-reader channel is easy, our attack can be very effective in dense reader environments where tags can be read multiple times. The relationship between our attack and Avoine attack is summarized in Table 7.1.

Type of Attack	No of rounds to be eavesdropped	No of guesses before revealing secret key
Avoine Attack	2	L^2
Our Attack	$\sqrt{\pi L}$ (on average)	1

Table 7.1: Comparison between our attack and Avoine attack.

7.5.3 Traceability Attack

We note an additional weakness of the *DIDRFID* protocol. If the final message C_i sent by the tag does not reach the reader due to a transmission error, or the attacker disrupts it, the reader does not recognize the updated value $DIDT_{i+1}$. The reader in this case asks for older values of $DIDT_i$ (this is not mentioned in [43]). In such a scenario, the attacker can track the tag by eavesdropping $DIDT_i, A_i, B_i$ and then disrupting message C_i. The attacker can then repeatedly ask for an older value $DIDT_i$ and send A_i, B_i in response, thus tracking the tag.

7.6 Conclusion

We have carried out security analysis of the two ultra-lightweight RFID authentication protocols, *SIDRFID* and *DIDRFID*, proposed by Yung-Cheng Lee [43]. Earlier analysis carried out by Avoine *et al.* [14] on *SIDRFID* mentions that the use of single master key is a potential weakness but does not describe the method to recover the master key. We have shown how to recover this single master key, thus allowing this weakness to be fully exploited. The attack on *DIDRFID* presented by Avoine *et al.* determines the correct key in L^2 attempts (where L is the length of key). We presented a full disclosure attack by reducing the number of attempts to $\sqrt{\pi L}$. We conclude that both *SIDRFID* and *DIDRFID* are extremely weak protocols.

Chapter 8

Conclusion

The integration of NFC with cellular technology has given a new dimension to the usage of NFC. It has a vast applications ranging from simple contents sharing to complex processes like m-commerce. NFC will be, most likely, the key technology that will be used in the mobile transaction in the coming years. In this context, the security of NFC is of crucial importance in success of NFC.

We focused on the security issues of NFC and RFID, mainly authentication. NFC and RFID tags are generally displayed in public with open access to any reader. For instance, smart posters that are used for advertisement purpose contain data that is accessible to any reader. The authentication of tag's contents is then becomes crucially important.

Similarly, there are occasions where copying the contents of an NFC tag to another tag is undesirable. For instance, an NFC tag used for access control, or an NFC tag registered against a specific product. In such cases, the authentication of the tag itself is very important.

Following the same approach, our contribution is subdivided into two main categories; Tag Data Authentication and Tag Authentication. The contributions are as follows:

1. NFC Forum released the Signature Specification in 2010 to digitally sign the tag's contents. The signature is computed over the *Type*, *ID* and *Payload* fields of an NDEF record, whereas the *lengths* fields and the NDEF header byte remain unsigned. The *partial* signing of NDEF message leads to multiple attacks such as Record Composition / Decomposition attack. These attacks exploit unsigned fields in the NDEF header. The earlier published attacks were not fully implementable as the required changes in the lengths fields were not taken into account. We refined these attacks and explained precisely what additional changes need

to be done. After refining the attacks, we provided two countermeasures to avoid such attacks.

The first countermeasure suggests for the inclusion of *TNF*, *Type-Length*, *Payload-Length* and *ID-Length* in the signature in such a way that the properties of an NDEF message, e.g., record chunking, remain intact. This requires a revision in the signature RTD only, without any alteration in the NDEF specification [63].

The second countermeasure is based on the removal of the data redundancy in the NDEF specification. Two bytes consisting of the *Type-length* and *ID-Length* fields are alway zero for the middle and terminating chunk records *(TNF=6)*. Hence, this is a redundant data as the *TNF* value in the header byte indicates the same. We suggested that removal of the lengths fields from the middle and terminating chunks (as these fields are always zero for chunk records) can fix some of the issues regarding signature specification [62].

We drew attention to our work by writing to the NFC Forum regarding the attacks and the proposed countermeasures. Consequently, the NFC Forum released a candidate specification for the updated Signature RTD Version 2.0 in 2013.

2. Presently, there is no mechanism provided by the NFC Forum to detect a counterfeit or cloned tag. This results in various possibilities for malicious activities where a legitimate tag is replaced by a counterfeit tag and the NFC tag reader is unable to detect the counterfeit. We proposed a framework to counter such attacks by providing a tag authenticating mechanism [64]. We introduced a new *Tag Authentication Record* that provides relevant information to authenticate a tag in an *off-line* environment. It employs public key cryptography with digital certificates and so can be used on NFC tags that have sufficient computational power and resources to perform such operations. The *Tag Authentication Record* is based on the NFC Data Exchange Format and is thus compatible with all NFC Forum devices. The NFC tag simply signs a challenge c and returns the signature to the NFC reader. The NFC reader verifies the signature according to the information available in the previously communicated *Tag Authentication* record. A successful verification confirms that the tag is not cloned. Of course, the certificate chain should also be checked for any revocation.

   Since the proposed specification requires public key encryption to be performed by the tag, the specification may be implementable to a small number of tags, at present, that have sufficient resources and power to perform public key cryptography. The scheme may be applied to more number of tags in future as their computational power increases over time. Moreover, light-weight versions of public key encryption schemes may also appear and allow wider applicability.

3. After proposing a mechanism to authenticate a tag, we proposed a framework to demonstrate its use in a supply chain to detect counterfeit tags [61]. We proposed that if the products are equipped with NFC tags, the counterfeit products can be identified by authenticating the attached NFC tags. During the initialisation phase, the product specifications like serial number, model, manufacturer, expiry date etc., are stored in the NFC tag along with the public key K_{pub} in the earlier proposed Tag Authentication Record. The data is signed according to the NFC Forum signature specification. An application in the user's cell phone reads the NFC tag's contents and detects the Tag Authentication record in the NDEF message stored in the tag. The user's phone executes a challenge response protocol to ascertain whether the tag has a correct private key K_{pr} (corresponding to the K_{pub}). A counterfeit product can be detected as it lacks the correct private key.

   We analyzed the economic aspects of the proposed scheme at a broader level as the inclusion of NFC tags require some additional investment by the supplier. Our analysis shows that the scheme is suitable for products with $30 or more profit margin (assuming 7% counterfeits in the market and $2/unit is the cost implementing NFC tag in a product).

4. Keeping in view that the main use of NFC in future will be commercial, we proposed a mobile payment solution [60]. The proposal is based on a cloud wallet model, where only the authentication credentials are stored in the mobile device and all sensitive information is stores in a cloud. Our protocol can be sued for the monetary transfer between two individuals who may not share the same mobile network. A customer, who wants to pay for some goods or services, is first authenticated by his MNO and after successful authentication, the amount is transferred to the shop account. We proposed a dedicated department, MNO Transaction Department (MTD), responsible for all the transactions. The security features of our protocol provide virtual tunnels among the mobile device, shop PoS terminal and the MTD. This caters for unsecured channels within various entities of the GSM network. The analysis shows the protocol is secure against various attacks by a dishonest customer or a dishonest shop.

5. I participated in the security analysis of two ultra-lightweight mutual authentication protocols, SIDRFID and DIDRFID [17]. The former is based on a static identity of RFID whereas the latter is based on dynamic identity of RFID. I contributed in a Full Disclosure Active (FDA) attack against SIDRFID. This is a

chosen plaintext attack where the attacker sends $L-1$ chosen public messages to the tag (L represents the length of the secret static tag ID). the chosen messages are designed in such a way that results in full disclosure of the secret.

8.1 Future of NFC

NFC can be used in variety of applications like identification, automation, ticketing, content sharing, payment, advertisement etc. In spite of its potential, standardised architectures are yet to develop to cater for its vast area of applications. Since NFC will be, most likely, used for m-commerce, the security is the back-bone of the NFC. Security issues emerging as a result of a sharp increase in its utility are a constant threat to the stability of an NFC ecosystem.

In addition to technical matters, there are concerns such as personalization, data storage, management and ownership of the Secure Element (SE) as it stores sensitive data such as banking credentials. Various key players in an NFC ecosystem, like mobile phone developers, financial institutions, smart card manufacturers, MNOs, operating system developers etc., are required to develop a widely acceptable NFC ecosystem. They need to improve their products in terms of compatibility and security. NFC, being a relatively new technology, is yet to mature to be widely used in m-commerce.

In a nutshell, the NFC would become a widely implemented main-stream technology provided that the NFC stake holders improve their products and standards to pull the NFC out of its current state of the Trough of Disillusionment. According to technical experts, this may take a time period of about two to five years.

Bibliography

[1] ETSI Specification of the Subscriber Identity Module - Mobile Equipment (SIM - ME) interface (GSM 11.11), December 1995. 89

[2] ISO/IEC 7816-4, Identification Cards – Integrated Circuit Cards, 2005. URL: http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=54550 [accessed: 2014-06-04]. 59

[3] NFC Data Exchange Format (NDEF): Technical Specification, July 2006. 32, 47, 48, 54, 64

[4] Smart Poster Record Type Definition: Technical Specification, July 2006. 32

[5] EPC Radio-Frequency Identity Protocols Class-1 Generation-2 UHF RFID Protocol for Communications at 860 MHz-960 MHz, 2008. 73, 113

[6] (X.509:) Information technology – Open Systems Interconnection – The Directory: Public key and attribute certificate frameworks, 2008. 60

[7] Signature Record Type Definition: Technical Specification, November 2010. 32, 54, 60, 68, 77

[8] PIN Transaction Security (PTS) Point of Interaction (POI), June 2013. URL: https://www.pcisecuritystandards.org [accessed: 2014-06-04]. 102

[9] Signature Record Type Definition: Technical Specification, April 2013. 33, 39, 52, 53

[10] G. Alpár, L. Batina, and R. Verdult. Using NFC Phones for Proving Credentials. In *The 16th International Conference on Measurement, Modelling, and Evaluation of Computing Systems and Dependability and Fault Tolerance, MMB & DFT 2012, Kaiserslautern, Germany*, volume 7201 of *Lecture Notes in Computer Science*, pages 317–330. Springer, 2012. 88

[11] A. Arbit, Y. Oren, and A. Wool. Toward Practical Public Key Anti-Counterfeiting for Low-Cost EPC Tags. In *International IEEE Conference on RFID*, pages 184–191, Orlando, USA, 2011. IEEE. URL:http://iss.oy.ne.ro/WIPR-IEEE. 58, 74

[12] A. Arbit, Y. Oren, and A. Wool. Toward Practical Public Key Anti-Counterfeiting for Low-Cost EPC Tags. In *International IEEE Conference on RFID*, pages 184–191, Orlando, USA, 2011. IEEE. URL:http://iss.oy.ne.ro/WIPR-IEEE. 71, 72, 74, 76

[13] P. Avery, F. Cerri, L. H. Fayle, K. Olsen, D. Scorpeci, and P. Stryzowski. The Economic Impact of Counterfeiting and Piracy. Organisation for Economic Co-operation and Development (OECD), 2008. 69

[14] G. Avoine and X. Carpent. Yet Another Ultralightweight Authentication Protocol that is Broken. In *Radio Frequency Identification. Security and Privacy Issues*, volume 7739 of *Lecture Notes in Computer Science*, pages 20–30, Nijmegen, Netherlands, 2013. Springer. 108, 112, 118, 121

[15] M. Bellare and C. Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In *Journal of Cryptology*, volume 21, pages 469–491. Springer, 2008. 103

[16] B. Berman. Strategies to Detect and Reduce Counterfeiting Activity. In *Business Horizons*, volume 51, pages 191 – 199. ELSEVIER, 2008. 69, 72

[17] Z. Bilal, K. Martin, and Q. Saeed. Multiple Attacks on Authentication Protocols for Low-Cost RFID Tags. In *Journal of Applied Mathematics and Information Sciences (To appear)*. Natural Sciences, 2014. 15, 106, 125

[18] A. Bouch. 3-D Secure: A Critical Review of 3-D Secure and its Effectiveness in Preventing Card Not Present Fraud. Master's thesis, Royal Holloway University of London, 2011. 104

[19] S. Canard, L. Ferreira, and M. Robshaw. Improved (and Practical) Public-Key Authentication for UHF RFID Tags. In *The 11th international conference on Smart Card Research and Advanced Application (CARDIS)*, volume 7771 of *Lecture Notes in Computer Science*, pages 46–61. Springer, 2012. 74

[20] L. Chao. What Happens When an eBay Steal Is a Fake. In *The Wall Street Journal*, June 2006. URL: http://online.wsj.com/news/articles/SB115154214225593742 [accessed: 2014-06-04]. 70

[21] W. Chen, G. Hancke, K. Mayes, Y. Lien, and J.-H. Chiu. NFC Mobile Transactions and Authentication Based on GSM Network. *International Workshop on Near Field Communication*, pages 83–89, 2010. 88

[22] W. Chen, G. Hancke, K. Mayes, Y. Lien, and J.-H. Chiu. Using 3G network components to enable NFC mobile transactions and authentication. *IEEE International Conference on Progress in Informatics and Computing (PIC)*, pages 441–448, 2010. 88

[23] H.-Y. Chien. SASI: A New Ultralightweight RFID Authentication Protocol Providing Strong Authentication and Strong Integrity. In *Transactions on Dependable and Secure Computing*, volume 4, pages 337–340. IEEE, December 2007. 107

[24] Cryptomathic. EMV Key Management Explained. White paper. URL: http://www.cryptomathic.com/media/51209/cryptomathic%20white%20paper-emv%20key%20management.pdf. 66

[25] J. de Ruiter and E. Poll. Formal Analysis of the EMV Protocol Suite. In *Theory of Security and Applications*, pages 113–129. Springer, 2012. 65

[26] ECMA-340. *Near Field Communication Interface and Protocol 1 (NFCIP-1)*. European Association for Standardizing Information and Communication Systems (ECMA), 2013. 22

[27] ECMA-352. *Near Field Communication Interface and Protocol 2 (NFCIP-2)*. European Association for Standardizing Information and Communication Systems (ECMA), 2013. 22

[28] W. Feller. Stirling's Formula. In *An Introduction to Probability Theory and Its Applications*, volume 1, pages 50–53. New York: Wiley, 1968. 119

[29] FirstData. EMV: A to Z (Terms and Definitions). Terms and definitions. URL: https://www.firstdata.com/downloads/marketing-merchant/EMV-A-toZ.pdf. 89

[30] L. Francis, G. P. Hancke, K. Mayes, and K. Markantonakis. On the security issues of NFC enabled mobile phones. In *International Journal of Internet Technology and Secured Transactions*, volume 2, Number 3-4/2010, pages 336–356. Inderscience Enterprises Ltd, 2010. 26

[31] L. Francis, G. P. Hancke, K. Mayes, and K. Markantonakis. Practical NFC Peer-to-Peer Relay Attack Using Mobile Phones. In S. B. O. Yalcin, editor, *RFIDSec*,

volume 6370 of *Lecture Notes in Computer Science*, pages 35–49. Springer, 2010. 27

[32] J. M. Gutiérrez, M. A. Hernández, P. J. Miana, and N. Romero. New identities in the Catalan triangle. In *Journal of Mathematical Analysis and Applications*, volume 341, pages 52–61. Elsevier, 2008. 119

[33] E. Haselsteiner and K. Breitfuß. Security in Near Field communication (NFC): Strengths and Weaknesses. In *Workshop on RFID Security (RFIDSec), July 12-14, Graz, Austria*, 2006. 26

[34] M. B. Hoy. Near Field Communication: Getting in Touch with Mobile Users. In *Medical Reference Services Quarterly*, volume 32, pages 351–357. Taylor & Francis, 2013. 23

[35] International Civil Aviation Organization (ICAO). Machine Readable Travel Document Document 9303, Part I, Vol II. `http://hasbrouck.org/documents/ICAO9303-pt1-vol2.pdf/`, 2006. 58, 59

[36] ISO/IEC 18092. *Information technology – Telecommunications and information exchange between systems – Near Field Communication – Interface and Protocol 1(NFCIP-1)*. International Organization for Standardization (ISO), 2013. 22

[37] ISO/IEC 21481. *Information technology – Telecommunications and information exchange between systems – Near Field Communication – Interface and Protocol 2(NFCIP-2)*. International Organization for Standardization (ISO), 2012. 22

[38] A. Juels. Strengthening EPC tags against cloning. In *Proceedings of the ACM Workshop on Wireless Security, Cologne, Germany*, pages 67–76. ACM, September 2005. 56

[39] A. Juels. RFID Security and Privacy: A Research Survey. In *Journal on Selected Areas in Communications*, volume 24, pages 381–394. IEEE, 2006. 107, 116

[40] A. Juels, D. Molnar, and D. Wagner. Security and Privacy Issues in E-passports. In *The First International Conference on Security and Privacy for Emerging Areas in Communications Networks*, pages 74–88. IEEE, September 2005. 57

[41] A. Juels and R. Pappu. Squealing Euros: Privacy Protection in RFID-Enabled Banknotes. In R. N. Wright, editor, *Financial Cryptography*, volume 2742 of *Lecture Notes in Computer Science*, pages 103–121. Springer, 2003. 56

[42] M. La Polla, F. Martinelli, and D. Sgandurra. A Survey on Security for Mobile Devices. In *Communications Surveys and Tutorials*, volume 15, pages 446–471. IEEE, March 2013. 82

[43] Y.-C. Lee. Two Ultralightweight Authentication Protocols for Low-Cost RFID Tags. In *Applied Mathematics and Information Sciences*, volume 6, pages 425–431. Natural Sciences, May 2012. 106, 107, 108, 112, 118, 121

[44] M. Lehtonen, J. Al-Kassab, F. Michahelles, and O. Kasten. Anti-counterfeiting Business Case Report. In *Technical report, BRIDGE Project*, December 2007. URL: `http://www.bridge-project.eu/data/File/BRIDGE_WP05_Anti_counterfeiting_business_case_report.pdf` [accessed: 2014-06-04]. 70, 71, 74

[45] M. Lehtonen, T. Staake, and F. Michahelles. From Identification to Authentication – A Review of RFID Product Authentication Techniques. In *Networked RFID Systems and Lightweight Cryptography*, pages 169–187. Springer Berlin Heidelberg, 2008. 55, 57

[46] G. Madlmayr and J. Langer. Near Field Communication Based Payment System. In *Konferenz Mobile und Ubiquitäre Informationssysteme (MMS), Germany*, volume 123 of *LNI*, pages 81–93. GI, 2008. 88

[47] G. Madlmayr, J. Langer, C. Kantner, and J. Scharinger. NFC Devices: Security and Privacy. In *Proceedings of the The Third International Conference on Availability, Reliability and Security, ARES, Barcelona, Spain*, pages 642–647. IEEE, 2008. 23, 24, 26

[48] D. Maimut and K. Ouafi. Lightweight Cryptography for RFID tags. In *Security & Privacy*, volume 10, pages 76–79. IEEE, 2012. 20

[49] C. Matlack and T. Mullaney. Fed Up With Fakes. In *Bloomberg Businessweek*, October 2006. URL: `http://www.businessweek.com/stories/2006-10-08/fed-up-with-fakes` [accessed: 2014-06-04]. 70

[50] Michael Roland and Josef Langer and Josef Scharinger. Security Vulnerabilities of the NDEF Signature Record Type. In *Third International Workshop on Near Field Communication, February 22-23, Hagenberg/Austria*, pages 65–70. IEEE, 2011. 38, 39, 41, 42

[51] NFC Forum. NFC Forum Tag Type Technical Specifications. 2010. 29

[52] Y. Oren and M. Feldhofer. WIPR–Public-Key Identification on Two Grains of Sand. In *Proceedings of RFIDSec*, volume 8, 2008. 64

[53] Y. Oren and M. Feldhofer. A Low-Resource Public-key Identification Scheme for RFID Tags and Sensor Nodes. In *Proceedings of the Second ACM Conference on Wireless Network Security, WISEC*, pages 59–68, Zurich, Switzerland, March 2009. ACM. 9, 58, 65, 74

[54] J. Paillès, C. Gaber, V. Alimi, and M. Pasquet. Payment and Privacy: A Key for the Development of NFC Mobile. In *International Symposiam on Collaborative Technologies and Systems*, pages 378–385. IEEE, 2010. 91

[55] A. Poschmann, M. J. B. Robshaw, F. Vater, and C. Paar. Lightweight Cryptography and RFID: Tackling the Hidden Overhead. In *Information, Security and Cryptology (ICISC)*, pages 129–145. Springer, 2009. 74

[56] P. Pourghomi, M. Saeed, and G. Ghinea. A Proposed NFC Payment Application. In *International Journal of Advanced Computer Science and Applications (IJACSA)*, volume 4, pages 173–181. SAI, 2013. 15, 92

[57] D. C. Ranasinghe, D. W. Engels, and P. H. Cole. Security and Privacy: Modest Proposals for Low-Cost RFID Systems. In *Auto-ID Labs Research Workshop, Zurich, Switzerland*, 2004. 57

[58] K. Rantos and K. Markantonakis. Analysis of Potential Vulnerabilities in Payment Terminals. In K. Markantonakis and K. Mayes, editors, *Secure Smart Embedded Devices, Platforms and Applications*, pages 311–333. Springer, 2014. 58, 65

[59] M. Roland and J. Langer. Digital Signature Records for the NFC Data Exchange Format. In *Second International Workshop on Near Field Communication, April 20-22, Monaco*, pages 71–76. IEEE, 2010. 35, 40, 42, 52

[60] M. Saeed, C. Walter, P. Pourghomi, and G. Ghinea. Mobile Transactions over NFC and GSM. In *The Eighth International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies, UBICOMM, Italy (To appear)*. IEEE, 2014. 15, 87, 125

[61] M. Q. Saeed, Z. Bilal, and C. D. Walter. An NFC based consumer-level counterfeit detection framework . In *The* 11^{th} *International Conference on Privacy, Security and Trust (PST 2013)*, pages 135–142. IEEE, July 2013. 14, 69, 125

[62] M. Q. Saeed and C. D. Walter. A Record Composition/Decomposition Attack on the NDEF Signature Record Type Definition. In *The* 6^{th} *International Conference for Internet Technology and Secured Transactions (ICITST-2011)*, pages 283–287. IEEE, Dec 2011. 14, 39, 43, 124

[63] M. Q. Saeed and C. D. Walter. An Attack on Signed NFC Records and Some Necessary Revisions of NFC Specifications. In *International Journal for Information Security Research (IJISR)*, pages 326–334, March 2012. 14, 39, 43, 68, 124

[64] M. Q. Saeed and C. D. Walter. Off-line NFC Tag Authentication. In *The 7th International Conference for Internet Technology and Secured Transactions (ICITST-2012)*, pages 730–735. IEEE, December 2012. 14, 54, 124

[65] T. Staake, F. Thiesse, and E. Fleisch. Extending the EPC network: the potential of RFID in anti-counterfeiting. In *Proceedings of the 2005 ACM Symposium on Applied Computing*, pages 1607–1612, USA, 2005. ACM. 20, 70, 74

[66] R. Stanley. *Enumerative Combinatorics*, volume 2. Cambridge Univ. Press, 1999. 119

[67] The Economist. The spread of counterfeiting: Knock-offs catch on. 2010. URL: http://www.economist.com/node/15610089 [accessed: 2014-06-04]. 70

[68] P. Tuyls and L. Batina. RFID-Tags for Anti-counterfeiting. In *The Cryptographers' Track, RSA Conference, San Jose, CA, USA*, volume 3860 of *Lecture Notes in Computer Science*, pages 115–131. Springer, February 2006. 57

[69] R. Want. An introduction to RFID technology. In *Pervasive Computing*, volume 5, pages 25–33. IEEE, Jan 2006. 18, 19, 20

[70] S. A. Weis, S. E. Sarma, R. L. Rivest, and D. W. Engels. Security and Privacy Aspects of Low-Cost Radio Frequency Identification Systems. In *Security in Pervasive Computing, First International Conference, Boppard, Germany*, volume 2802 of *Lecture Notes in Computer Science*, pages 201–212. Springer, 2003. 56

Appendix A

NFC-Enabled Mobile Phones

Acer Cloud Mobile	Acer E320 Liquid Express
Acer Liquid Glow	Acer Liquid S2
Adlink IMX-2000	Alcatel One Touch 922
Alcatel One Touch 996	Alcatel Onetouch Idol 2
Alcatel Onetouch Idol 2 Mini S	Alcatel Onetouch Idol 2S
Alcatel Onetouch Pop Fit	Asmaitha Sruta 7 Tablet
Asus Padfone 2	Asus Padfone Infinity
Asus Vivo Tab	Asus Vivo Tab RT
Asus VivoTab Smart	BBK Vivo Xplay
Benq T80	BlackBerry Bold 9790
BlackBerry Bold 9900/9930	BlackBerry Curve 9350/9360/9370
BlackBerry Curve 9380	BlackBerry PlayBook
BlackBerry Q10	BlackBerry Q5
BlackBerry Z10	BlackBerry Z30
BWC ToughSlate 7	C-Mii 1
C-Mii 3	Casio DT-X8
Casio GzOne CA-201L	Casio IT-800
Cetrix CB250	Cetrix CD661
Cetrix CT973G	Cetrix CV300
Dell Venue 11 Pro	DLI 9000
Faea F1	Faea F2
Faea F2S	Firefox OS Flame
Fujitsu Arrows A	Fujitsu Arrows F-07D
Fujitsu Arrows Kiss	Fujitsu Arrows Tab
Fujitsu Arrows V	G.To N800

Gentag GT-601v2
Gionee Elife E7
Google Nexus 107
Google Nexus 5
Google Nexus 7 (2013)
Hike X1
Hike X1D
Hisense Sero 7 Pro
HP Elitebook Revolve
HP Elitepad 900
HTC Desire 500
HTC Desire 610
HTC Desire 816
HTC Desire C
HTC Droid DNA/HTC J Butterfly
HTC Droid Incredible 4G LTE
HTC Evo 4G LTE
HTC First
HTC Incredible
HTC Mini
HTC One
HTC One M8
HTC One Max
HTC One SV
HTC One VX
HTC One X/XL
HTC Ruby/Amaze 4G
HTC Windows Phone 8X
Huawei Ascend G300
Huawei Ascend G6 4G
Huawei Ascend G600
Huawei Ascend P2
Huawei Ascend Y201
Huawei Sonic/Turkcell T20
Huawei TalkBand B1
iBerry Auxus Nuclea N2
Jolla by Jolla
Kuoziro FT701W NFC Tablet
Kyocera Hydro Elite
Kyocera Torque
Lenovo K800
Lenovo ThinkPad Tablet 2
LG G Flex
LG G Pro 2
LG G2
LG G3
LG KU380-NFC
LG Mach
LG Optimus 3D Max
LG Optimus 4X HD
LG Optimus Elite
LG Optimus G
LG Optimus L5
LG Optimus L7
LG Optimus LTE
LG Optimus LTE Tag
LG Optimus Net
LG Optimus Vu
LG T530 Ego
LG Viper
Lumigon T2
Lumigon T2 HD
M3 Android NFC Communicator
Megafon Mint
Meizu MX3
Motorola Droid Maxx
Motorola Droid Mini
Motorola Droid Razr
Motorola Droid Razr HD
Motorola Droid Razr M
Motorola Droid Razr M 4G LTE7
Motorola Droid Razr Maxx HD
Motorola Droid Ultra
Motorola MC75A HF

Motorola Moto X
Motorola Photon Q 4G LTE
Motorola Razr D3
Motorola Razr i/MT788
MTS 975
Nokia 603
Nokia 700
Nokia 701
Nokia 801T
Nokia 808 PureView
Nokia C7/Astound
Nokia Lumia 1020
Nokia Lumia 1520
Nokia Lumia 2520
Nokia Lumia 610 NFC
Nokia Lumia 620
Nokia Lumia 720
Nokia Lumia 820
Nokia Lumia 920
Nokia Lumia 925
Nokia Lumia 928
Nokia Lumia 930
Nokia Lumia Icon
Nokia N9
Nokia Oro
OnePlus One
Oppo Find 5
Oppo Find 7
Oppo N1
Orange Infinity 996
Orange San Diego
OrientPhone P6 Plus
Panasonic BizPad
Panasonic Eluga
Panasonic Eluga Power7
Pantech Discover
Pantech Sky Vega LTE
Pantech Sky Vega Racer
Philips Xenium W336
Porsche Design P9981
Porsche Design P9982
Prada phone by LG 3.0
Samsung Ativ Odyssey
Samsung Ativ S Neo
Samsung Ativ SE
Samsung Galaxy Ace 2
Samsung Galaxy Ace Style7
Samsung Galaxy Axiom
Samsung Galaxy Core Advance
Samsung Galaxy Core LTE
Samsung Galaxy Express
Samsung Galaxy Express 2
Samsung Galaxy Light
Samsung Galaxy Mega
Samsung Galaxy Mini 2
Samsung Galaxy Note
Samsung Galaxy Note 3
Samsung Galaxy Note II
Samsung Galaxy Premier
Samsung Galaxy Round
Samsung Galaxy Rugby LTE/Pro
Samsung Galaxy S Advance
Samsung Galaxy S Blaze 4G
Samsung Galaxy S II
Samsung Galaxy S II Plus
Samsung Galaxy S III
Samsung Galaxy S III Mini
Samsung Galaxy S4
Samsung Galaxy S4 Active
Samsung Galaxy S4 Mini
Samsung Galaxy S5
Samsung Galaxy Stratosphere II
Samsung Galaxy Victory 4G LTE
Samsung Galaxy Young

Samsung S5230 NFC7
Samsung S5260 NFC
Samsung SHW-A170K
Samsung Wave 578
Samsung Wave M7
Samsung Wave Y
Samsung Windows RT Ativ Tablet
Samsung WP8 Ativ S
Samsung Z
Sharp Aquos Phone Serie
Sharp Aquos Phone Zeta
Sharp RW-T107 NFC Tablet
Sharp RW-T110 NFC Tablet
Sonim XP1301 Core NFC
Sonim XPand NFC
Sony SWR10 SmartBand
Sony Vaio Fit
Sony Xperia Acro S
Sony Xperia AX
Sony Xperia Ion
Sony Xperia L
Sony Xperia M
Sony Xperia M2
Sony Xperia P
Sony Xperia S
Sony Xperia Sola
Sony Xperia SP
Sony Xperia T
Sony Xperia T2 Ultra
Sony Xperia Tablet Z
Sony Xperia V
Sony Xperia VL
Sony Xperia Z
Sony Xperia Z1
Sony Xperia Z1 Compact
Sony Xperia Z2
Sony Xperia Z2 Tablet7
Sony Xperia ZL
Sony Xperia ZR
TazTag TazPad7
TazTag TPH-One7
The Toughphone Defender
Toughshield R-500
Toughshield T700
Turkcell MaxiPRO5
Turkcell T11/ZTE Racer II
Turkcell T40
Umi Cross
Umi X2S
Vertu Constellation
Vertu Ti
Vodafone Smart III
Xiaomi Mi 2A7
Xiaomi Mi3
Xolo X900
Yota Devices YotaPhone (2014)7
Yulong Coolpad 8870 NFC7
Zopo ZP998
ZTE Blade II
ZTE GoTa GH800
ZTE Grand X IN
ZTE Kis
ZTE Nubia Z5
ZTE Orbit
ZTE PF200
ZTE R233
ZTE Turkcell MaxiPLUS5