B.Sc. in Computer Science and Engineering Thesis

NFC Relay Attack & Prevention

Submitted by

Asif Sanjary
201414043

Jahidul Islam
201414020

Mehedi Afzal Farazi
201414045

Supervised by

Shohrab Hossain

Associate Professor

Computer Science & Engineering

Bangladesh University of Engineering & Technology (BUET)

Department of Computer Science and Engineering
Military Institute of Science and Technology

December 2017

CERTIFICATION

This thesis paper titled **"NFC Relay Attack & Prevention"**, submitted by the group as mentioned below has been accepted as satisfactory in partial fulfillment of the requirements for the degree B.Sc. in Computer Science and Engineering in December 2017.

Group Members:

1. **Asif Sanjary**
2. **Jahidul Islam**
3. **Mehedi Afzal Farazi**

Supervisor:

Shohrab Hossain
Associate Professor
Computer Science & Engineering
Bangladesh University of Engineering & Technology (BUET)

CANDIDATES' DECLARATION

This is to certify that the work presented in this thesis paper, titled, "NFC Relay Attack & Prevention", is the outcome of the investigation and research carried out by the following students under the supervision of Shohrab Hossain, Associate Professor, Computer Science & Engineering, Bangladesh University of Engineering & Technology (BUET).

It is also declared that neither this thesis paper nor any part thereof has been submitted anywhere else for the award of any degree, diploma or other qualifications.

_____

Asif Sanjary
201414043

_____

Jahidul Islam
201414020

_____

Mehedi Afzal Farazi
201414045

ACKNOWLEDGEMENT

We are thankful to Almighty Allah for his blessings for the successful completion of our thesis. Our heartiest gratitude, profound indebtedness and deep respect go to our supervisor, **Shohrab Hossain**, Associate Professor, Department of Computer Science & Engineering, Bangladesh University of Engineering & Technology, for his constant supervision, affectionate guidance and great encouragement and motivation. His keen interest on the topic and valuable advice throughout the study was of great help in completing thesis.

We are especially grateful to the Department of Computer Science and Engineering (CSE) of Military Institute of Science and Technology (MIST) for providing their all out support during the thesis work.

Finally, we would like to thank our families and our course mates for their appreciable assistance, patience and suggestions during the course of our thesis.

Dhaka
December 2017.

1. Asif Sanjary
2. Jahidul Islam
3. Mehedi Afzal Farazi

ABSTRACT

Contactless Smart Cards are a new addition to the technology world. They have paved the way to payment, authentication and access. Smart cards are vulnerable to a relay attack where the card is read e.g. from the victims pocket and used remotely for unauthorized purposes. NFC compliant Smart Cards are often a victim to Relay Attack. This thesis project has two goals: (1) to implement relay attacks against NFC smart cards, such as transport tickets based on Mifare DESFire EV1 technologies, and (2) to investigate how time measurements could be used to detect and prevent such attacks. We have utilized an app to read the memory of Mifare Classic 1K cards. The app is able to dump all the data from memory of an NFC TAG. Finally we proposed counter-measures to relay attack such that the round-trip time delay can help to detect the attack and terminate communication.

Contents

List of Figures

LIST OF ABBREVIATION

NFC : Near Field Communication
PCD : Proximity Coupling Device
PICC : Proximity Integrated Circuit Card
APDU : Application Protocol Data Unit
FWT : Frame Waiting Time
HCE : Host Card Emulation
POS : Point of Sale
PC : Personal Computer
OS : Operating System
MITM : Man-in-the-middle Attack
iCVV : Dynamic Card Verification Value
UUID : Universally Unique Identifier
qVSDC : quick Visa Smart Debit/Credit
ATS : Answer to Select

CHAPTER 1
INTRODUCTION

Near Field Communication (NFC) technology exists in our daily life and has integrated into many aspects of our activities(See Chapter 2). Originally developed as a subset of RFID technology, NFC features a low range but high frequency wireless communication between two devices. Contactless smart cards, or say, NFC smart cards are nowadays used in many applications including ticketing system like bus tickets, payment systems like NFC embedded Master card [1] and even identification cards which can unlock the door. Already there have been plenty of researches on the security features of NFC applications [2–7].

The emergence of NFC technology is far earlier and tools to operate NFC are dedicated devices from vendors. In the past, business solution programmers or researchers have the resource to explore the pattern of NFC through dedicated devices and proprietary protocols. Normal users possessing NFC cards can only check their information of cards and read or write information through official provided service points. For example, the bus card users can only check their balance in cards via top-up machines. However, the situation has changed since smart phones with NFC chips come into common users life. A normal user has no knowledge of NFC specifications can read/write his card and even customize his NFC tag with the help of smart phone applications.

With the unveiling of the NFC technology, a piece of hardware enabled NFC features can easily perform the following two important tasks in NFC, read/write mode and card emulation mode. As the NFC technology is labeled as a low distance and encryption protected communication, we took a relentless approach to perform relay attack [8].

1.1 Problem Statement

John leaves home in the morning with his bus travel card on the desk with a smart phone attached to it. When he catches up the bus and trying to pay for the trip, instead of showing his travel card, he shows another smart phone from his pocket to the reader. Surprisingly, the reader accepts it and beeps as normal. The bus driver cannot believe his eyes and thinks it must be some dark magic. What he does not know is that Johns card at home updates the value simultaneously already via cellular network by the smart phone put on it. John does

not cheat, all he does is a simple relay attack on NFC technology

Our goal in this thesis research is to prevent relay attack on NFC smart cards, Forcing real contactless smart card to be physically presented to service points, e.g. bus card to bus card reader. Traditionally an attacker who is performing a relay attack needs to sit in between two communication parties. As we already know the specification of NFC is to communicate in an extremely low distance by 1-10cm, attackers cannot derive a possible relay attack without enlarging this distance. With the help of NFC hardware operating in card emulation mode, we can already impersonate the interface of a real card [8]. Connected via a third channel, another piece of NFC hardware can read/write information accordingly to the real card somewhere else. Thus, the attacker can fully hijack the communication between reader and card and sniff the traffic.

As nowadays NFC chips are becoming a standard part in smart phones, it lowers the difficulty to get a useful device to perform card emulation. Our research would use a modified Android operating system to control the NFC chip in smart phone over certain level. Standard Android ROM does not provide us APIs to emulate NFC tags freely, only modified Android can do emulation. One smart phone is used to impersonate the card and the other is going to read/write transactions to the real card. Transactions are recorded for further understanding. Communication between two smart phones is via Wi-Fi or Bluetooth channel. This communication channel covers a far distance of 46m indoor or 92m outdoor if using Wi-Fi 802.11, and 31m for Bluetooth 2.1 [9].

On our approach towards Relay Attack, we successfully read the memory of an NFC Tag. We did that by building an app specifically for our NFC chip. The app dumps the the data on the memory.

Next we propose a prevention theory which will prevent Relay Attack on NFC payment transaction. Our proposal is based on the time delay on round-trip time, which can be detect by setting up two counters on both reader and tag.

1.2 Thesis Organization

In Chapter 2, we present the background on NFC, the protocols NFC use and different attacks on NFC and previous prevention proposals on Relay Attack..

In chapter 3, we discuss in detail how Relay Attack is done, what are the components needed for Relay Attack and how we theorized our own prevention proposal.

In chapter 4, we described how we read data from TAG memory. We discussed the work flow of our App and finally we conclude it with some important functions of the app.

In chapter 5, we discuss our dependencies and limitations. We also describe how we will move toward with our thesis. As we are not going to abandon this topic, because it is a very advanced topic with lots of things to consider.

At last in the conclusion, we concluded our thesis work.

CHAPTER 2
BACKGROUND AND PREVIOUS STUDIES

Wireless technologies have made lives more comfortable and every tasks more convenient. From the end times of 20th century till now more and more research have been done in wireless technologies. These fields have evolved so much that nowadays we dont want to plug or swipe anything. We just want everything done in the air.

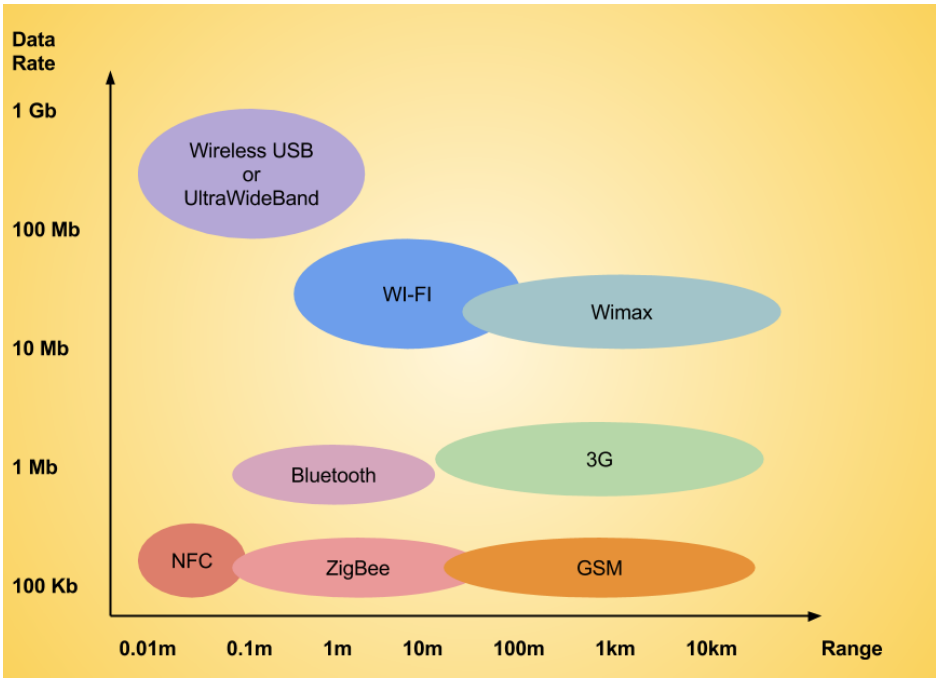

Figure 2.1: NFC and Other Wireless Technologies

GPRS-EDGE, 3G, 4G, Wimax, Wi-Fi etc. are connecting us to the whole world through voice calls, messages, video calls and other internet facilities. These technologies mainly focus in long distance communications. On the other hand Bluetooth, Zigbee, Infrared Transmission, NFC etc. are short range wireless communication. With these technologies we can transfer files between devices, connect two devices, do home automation, create wireless sensor networks, do payment, authentication etc. Among them NFC has the maximum short range which is less than 20cm. And its ideal for wireless security applications as it seems that no other device can have a peek on that little range. Therefore NFC has uses where proximity is needed.

With NFC, a smartphone can be used to behave like a credit or debit card and without carrying a card, payments can be done. Right now Masterpass lets its consumers store credit/debit card info and consumers can use Masterpass Tap & Pay feature which is NFC actually in stores where NFC Payment is supported. Google wallet, Samsung Pay and Apple Pay provide the same feature.

NFC has other uses too. Two phones can beam content like contacts, apps, photos, videos etc. at each other. And then there are NFC tags which have many uses like - smart home, smart car, settings tasks like - wake up, bed time, turn on-off computer remotely, business cards, access to Wi-Fi and bluetooth, make phonecall, open apps, open URL etc.

NFC is mainly known for payment and there have been more researches on the vulnerabilities of the payment transaction such as - eavesdropping, data modification, relays etc. But only Relay Attack on NFC payment is more feasible than the other two. And there have been propositions on how the Relay attack can be prevented but the protocol that NFC is based on is more of lightweight to carry that much burden to prevent Relay attack.

2.1 NFC

NFC connects two devices where both or one generates own RF field to communicate with each other. With NFC communication, the transfer speed is 106, 212 or 424 kbps. NFC operates on frequency 13.56 MHz.

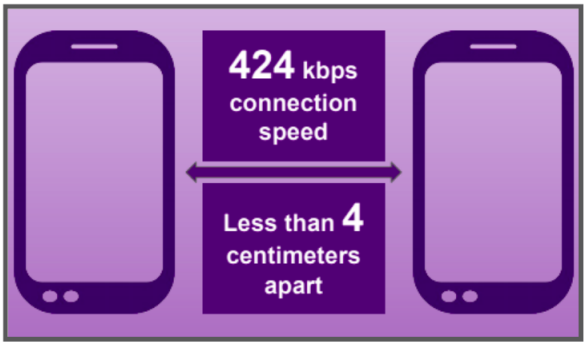

Figure 2.2: NFC specification

Two devices that are connected with NFC, One of them is called PCD (Proximity Coupling Devices) and the other is PICC(Proximity Integrated Circuit Card).

NFC devices communicate using electromagnetic induction. As one of the two devices in NFC communication generates a magnetic field which in turn powers the other device if that device has no own power supply.

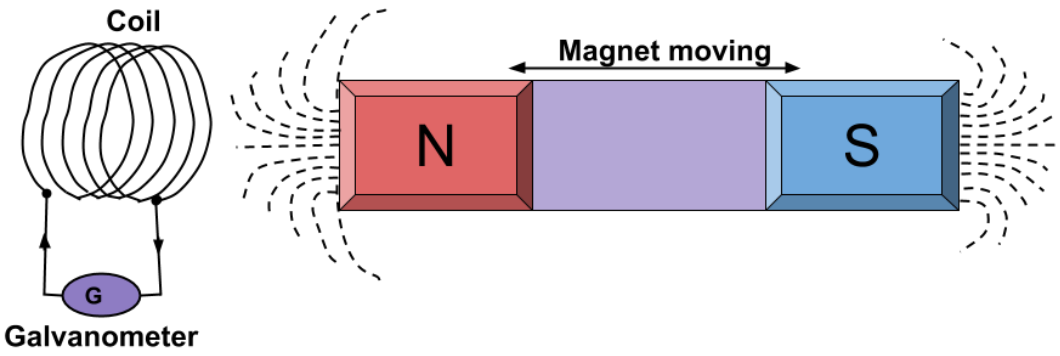

Figure 2.3: How NFC works

NFC operation has three modes. Reader-Writer, Peer to Peer and Card-emulation. In reader-writer mode, data is read or written such as URL, Coupons etc. Peer to Peer mode is the communication between two Reader-Writer NFC devices such as smartphones. When an NFC device acts as a contactless smart card, its called Card-emulation mode. In this mode the smartphone can do contactless payments, ticketing, access-control etc.

If both devices generate their own RF field, then this is called Active Communication. Active Peer to Peer and Card-Emulation modes are of this type. Here both devices when trying to communicate, modulates its field with Amplitude Shift Key(ASK) and the other device turns off its field to avoid collision.

And when PCD generates the RF field and PICC modulates the field to communicate, this is called Passive Communication. Read-Write, Passive Peer to Peer etc. are of this type. In active communication both devices have power supply and in passive communication PCD has power supply and PICC may draw power from the electromagnetic field generated by PCD.

2.2 NFC & other Technologies

Topic	RFID	NFC	Bluetooth	Zigbee
Frequency Band	125 - 134 KHz (LF) 13.56 MHz (HF) 856-960 MHz (UHF) 2.45 GHz 5.8 GHz	13.56 MHz	2.4GHz	2.4GHz
Communication Distance	10 cm (LF) 30 cm (HF) 100m (UHF)	10cm	~10m	10 to 20m
Modulation Method	ASK, FSK, PSK	ASK	FSK, PSK	OQPSK
Communication Mode	Active to Active Active to Passive	Active to Active Active to Passive	Active to Active	Active to Active
Setup Time	varies	<0.1s	~6s	>1s
Application	Tagging & tracking of goods and items for manufacturing, logistics, retail, etc	Smartphones, tablets, portable devices in a peer-to-peer network	Smartphones, tablets, audio equipment, printers, other devices in a personal area network (PAN)	Lighting networks, home automation, industrial control

Figure 2.4: NFC and Other Short Range Wireless Technologies

2.3 ISO 14443

Contactless smart card manufacturers have their own proprietary protocol. But all NFC device manufacturers must implement ISO 14443 [10] NFC protocol stack to be compatible with other NFC devices for different NFC applications.

ISO/IEC 14443 is an international standard which determines how the NFC supported cards can be used for access control and the communication between the card and the reader.[1] If we want to consider from top to bottom, NFC enabled smart phones use a protocol stack that consists of ISO 7816-4 [11], ISO 14443-4 [12], ISO 14443-3 [13], ISO 14443-2 [14] ans ISO 14443-1 [15] protocols. Here we will briefly discuss these protocols.

ISO 14443-1 describes the physical characteristics of the Proximity Integrated Circuit Cards (PICC). The physical characteristics are such as the antenna dimensions and location, the magnetic field properties etc.

ISO 14443-2 defines the modulation, subcarrier, data rate, bit representation for communication of both ends, PCD and PICC. There are two defined interfaces Type-A and Type-B.

Above it, there is ISO 14443-3. When PICC enters into the field of PCD, this is an initiali-

sation phase between PICC and PCD which describes the timing, byte formats and frames. And if there are more than one PICC in the field of PCD, PCD selects the correct PICC among others with anticollision phase.

ISO 14443-4, a transmission protocol featuring data block exchange and multi-activation mechanisms to enable successful transmission of messages between PICC and PCD where visible appearance of messages transmitted is in bytes and quite raw.

And at the top is ISO 7816-4 and it is independent of the physical characteristics. It mainly defines the access methods and security to access rights of files and data in the card. This layer helps programmers to be less troubled with exploring raw data in bytes by encapsulation and abstraction of the message format.

2.4 Current Credit Card Protocol

The protocol for performing contactless credit card trans- actions over NFC (termed CC Protocol) is composed of two query-response pairs. First, the NFC reader (hereafter referred to as a Point-of-Sale) solicits the card for its credit card number and expiration date, and the card responds with this information. In its response, the credit card also includes an iCVV [16], or integrated Card Verification Value: a dynamic security token intended to authenticate the message. Once this has completed, the Point-of-Sale sends a charge request to the bank with the information received from the credit card, and then receives an authorization response to accept or reject the charge.

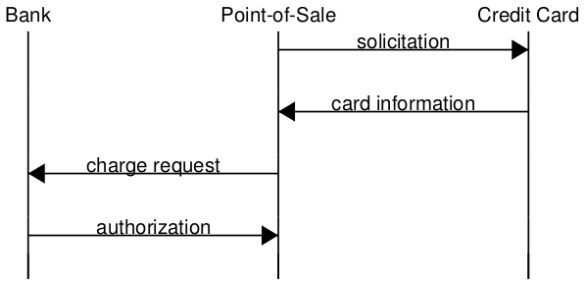

Figure 2.5: The current CC Protocol

The exchange of messages in the CC Protocol is shown in Figure 2.5. They are: solicitation, card information, charge request and authorization. Note that after the card responds to the Point-of-Sale, its involvement in the transaction is complete. The contents of these messages are as follows:

2.4.1 Solicitation

First, the Point-of-Sale solicits the credit card for its information. The solicitation is composed of a number of messages sent in both directions, identifying the Point-of-Sale type (e.g. 2PAY.SYS.DDF01) and the credit card type (e.g. VISA CREDIT). Since these messages are constant for a given Point-of-Sale and credit card, we abstract the solicitation messages as a single request from the Point-of-Sale to the credit card.

2.4.2 Card Information

The credit card responds to the solicitation by sending back the following card information:

- the credit card number
- the credit cards expiration date
- the iCVV
- the name of the bank that issued the card

The iCVV is an unpredictable 4-byte value freshly generated for every solicitation response, and is subsequently used by the bank to validate the transaction as described below.

2.4.3 Charge Request

The Point-of-Sale issues a charge request to the bank. This request is composed of:

- the credit card number
- the credit cards expiration date
- the iCVV
- the dollar amount to be charged

2.4.4 Authorization

When the bank receives a charge request, it uses the credit card number to look up the account, verifies the expiration date, and then validates the iCVV to authorize the purchase. It will generally also perform some additional checks, such as verifying that the card was

not reported lost or stolen, or matching this purchases location against the known location of the card holder. Finally, it responds with its authorization decision.

When the credit card is manufactured, a secret seed value is shared between the credit card and the bank. This enables the credit card and the bank to both generate the same iCVV sequence, unpredictable to any party that does not have access to this seed. The iCVVs are simply sequential elements of this sequence : each time the credit card responds to a solicitation, it generates the next iCVV in the sequence and includes it with the card information response.

In order to make an authorization decision, the bank searches through its account database, which is indexed by the credit card number. Once the bank locates the account, it verifies that the received expiration date matches the expiration date on file. In addition, it recalls the iCVV from this credit cards previous charge request and generates the next element in the sequence, then compares the received iCVV to the value it generated.

It is possible that a card may generate an iCVV without communicating it to the bank. For example, a charge request may become corrupted in transit, or a Point-of-Sale may experience a network failure. As a result, a credit cards iCVV may have advanced further in the sequence than the bank expects. To handle this situation, the bank may generate several iCVVs in the sequence for comparison to the received value.

If a match is found, the bank considers the iCVV to be valid. It updates its state into the pseudo-random sequence to reflect the received iCVV, and continues with any other checks to be performed before authorizing the charge. If no match is found, the bank considers the iCVV to be invalid and declines the charge.

2.5 Attacks on NFC

There are manily three kinds of attacks on NFC, such as - Eavesdropping, Data Modification and Relay Attack.

2.5.1 Eavesdropping

Eavesdropping on NFC communication might be hard as magnetic coupling reduces the readers signal travel distance and so the maximum distance from sender to interceptor is reduced as the interceptors antenna needs to be very close to receive the signal [17]. Diakos et al. [18] performed an eavesdropping experiment on NFC payment.

2.5.2 Data Modification Attack

In Data Modification attack scenario, the data being exchanged is captured and modified by an attackers radio frequency device. The attackers device is able to inhibit the NFC data exchange briefly, but long enough to alter the binary coding. This type of attack is very difficult to implement but the data modification is realizable in rare cases, especially for active mode transmission of NFC information. The most common way to interfere with the NFC data exchange is to use an RFID jammer; Data modification could be detected, introducing code in the NFC source device that measures the strength of frequencies, thus choosing the one that is truly the closest and most likely valid. Checking the RF field during transmission allows the sender to detect this type of attack. Another possibility is to modify the data in such a way that it appears to be valid to the receiver; the attacker has to deal with the single bits of the RF signal. The feasibility of this attack depends on various factors, such as the strength of the amplitude modulation. As described in Mostafa [19], transferring data with modified Miller coding and a modulation of 100%, only certain bits can be modified, while transmitting Manchester-encoded data with a modulation ratio of 10% permits a modification attack on all the bits.

2.5.3 Relay Attack

A relay attack exploits the ISO/IEC14443 protocol compliance of NFC; the attacker has to forward the request of the reader to the victim and relay back its answer to the reader in real time in order to carry out a task by pretending to be the owner of the victims smart card.

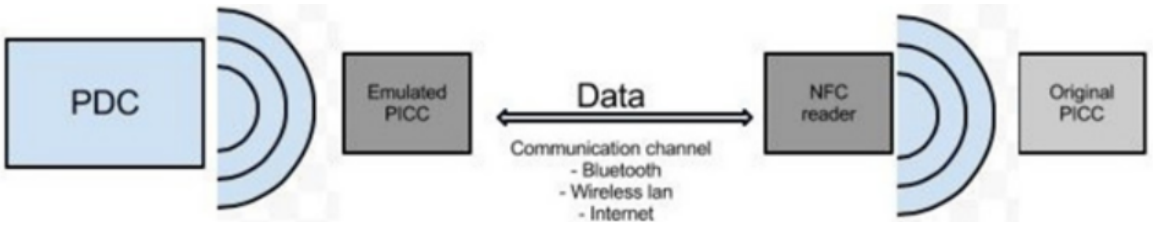

Figure 2.6: Relay attack scheme

This attack technique focuses on the extension of the range between the NFC token (e.g., a card) and the reader to implement it two NFC enabled devices are necessary, one acting as a reader and one acting as a card emulator. The access victim system will not able to detect the attack because it will think a card is actually in front of it.

In the attack scenario the attacker holds the NFC reader near the victims card and relays the data over another communication channel to a second NFC reader placed in proximity to the original reader that will emulate the victims card.

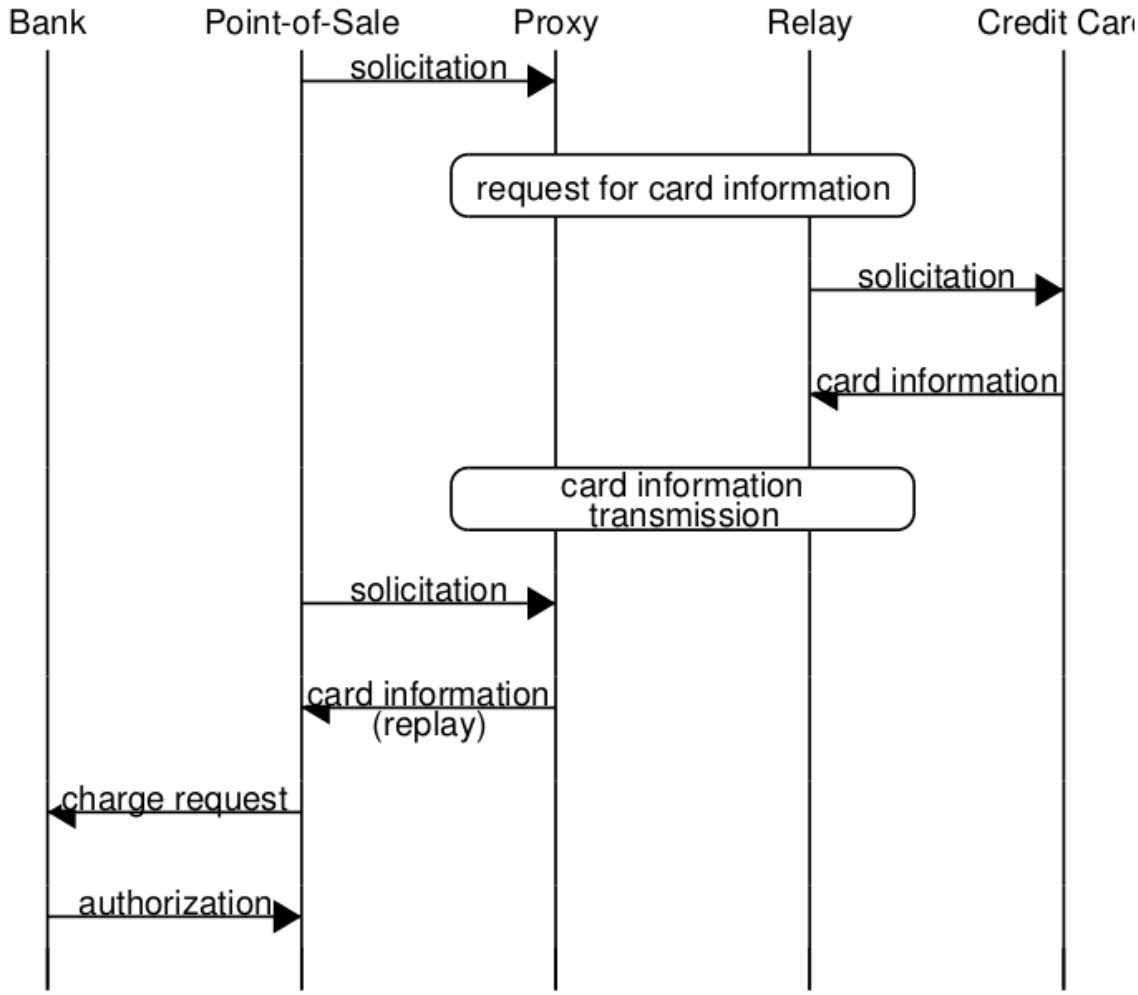

Figure 2.7: Simple Diagram of Relay Attack on Payment Transaction

The attack is constrained by a timing issue: Because of the physical distance between the two NFC devices, the packets that are relayed will take longer to be transferred to the destination.

RFID technology has some constraints on the time range between a challenge and response, named frame waiting time (FWT); exceeding this limit will cause the failure of the attack.

2.6 Previous Prevention Proposals

There have been proposals on preventing Relay Attack on NFC. Following notable proposals are given.

2.6.1 Protocol Modification

OLiver, Mohamed and Lili proposed a new protocol for contactless credit card transaction [20]. They call it ”Secure CC Protocol”. As we see in the figure2.8 the four mes-

sages have been modified to make the protocol secure which defends against Eavesdropping, Skimming, Relay Attacks and Compromised Point-of-Sale. Let's first understand the four messages.

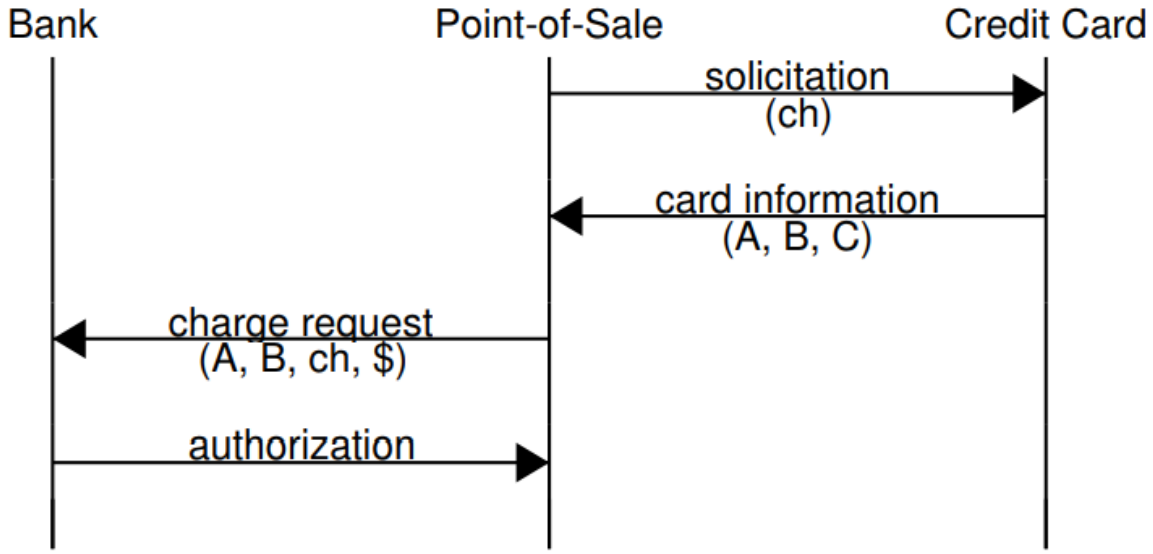

Figure 2.8: Proposed Secure CC Protocol

Solicitation*(ch)***:** The Point-of-Sale sends a solicitation message much like in the current CC Protocol. However, this message now includes a random challenge *ch*. This challenge is used by the credit card when constructing its response.

Card Information*(A, B, C)***:** The solicitation response consists of three components, A, B and C. They are as follows:

A = UUID, a Universally Unique Identifier, is used to identify the card without revealing the cards information to eavesdroppers or any other party. Our use of a UUID, which requires a significant source of entropy, does not violate our computational constraints by the fact that it is fixed. As such, this value is computed by the credit card manufacturer and stored on the card as a constant.

B = H(info, ch, iCVV) is used to authenticate the cards identity. This abstract function H will be defined later through stepwise refinement, but informally we can think of it as similar to a cryptographic hash function: it can be used to verify its arguments while leaking no information about them.

C = bank name is transmitted in the clear, as before, so that the Point-of-Sale may route its charge request to the proper entity.

In so doing, A provides identification, B provides authentication, and C provides routing information. In addition, A can be used by retailers in order to recognize credit cards, allowing retailers to continue the popular practice of correlating purchases.

Charge Request *(A, B, ch, $)*: The Point-of-Sale issues a charge request to the credit cards issuing bank. In this protocol, the Point-of-Sale does not learn the cards private information, but simply forwards the card identification (A) and authentication (B) to the bank specified by C. The Point-of-Sale also includes the challenge *ch* so that the bank can verify that B is a valid response to ch from card A, as well as the dollar amount to be charged.

Authorization: The bank maintains an index of UUID into its account database. When the bank receives a charge request, it identifies the matching record as specified by A, looking up $info_{bank}$ and $iCVV_{bank}$. It then calculates $B_{bank} = H(info_{bank}, ch, iCVV_{bank})$ and verifies that $B = B_{bank}$. This step is equivalent to verifying the credit card information and *iCVV* in the current CC Protocol, resulting in an authorization decision to accept or reject the charge request.

The secure CC protocol was defined in terms of an abstract function H. If function H satisfies two properties namely *H1* and *H2*, then the CC protocol isn't vulnerable to four classes of attacks. And function H can be redefined in terms of function F and G. Function F must satisfy properties *F1* and *F2*. Again function G must satisfy property *G1*. And if all properties are satisfied only then CC protocol isn't vulnerable.

$H(info, ch, iCV\ V\ ) = F(x, iCVV\ )$ where $x = G(info, ch)$

Here property *G1* is that two info of different credit cards can't be same and if someone even has the info, one can't infer x without knowing the *ch* which is a random challenge.

F1 is satisfiable only when *iCVV* is indistinguishable from random.

F2 is satisfiable only when one can't infer F just by knowing $x = G(info, ch)$ or *iCVV*.

And property *H1* says that if *iCVV* is indistinguishable from random, then $H(info, ch, iCV\ V\ )$ is also indistinguishable from random.

H2 says that two random challenges *ch* can't be same and just by knowing *ch*, one can't infer $H(info, ch, iCV\ V\ )$.

Here from the Figure 2.9 we can see the implementation of the secure cc protocol.

```
function G(info, ch):
    const khi = hash(bank_key, info)
    result = empty list of bits
    for each of the n bits of ch:
        if the n'th bit of ch is 1:
            append n'th bit of khi to result
    return result

function F(x, iCVV):
    return x XOR iCVV

function H(info, ch, iCVV):
    x = G(info, ch)
    return F(x, iCVV)
```

Figure 2.9: Proposed implementation of F, G and H

Here *khi* is constant which is the output of $h_k(info)$. The function h_k is a strong cryptographic hash function keyed with the banks secret key k. As *info* does not change over the lifetime of the credit card, $h_k(info)$ is stored as a constant on the card. As such, the computation necessary for hashing is not executed on the credit card, and the card requires no knowledge of the banks secret key k.

Their implementation also requrired that $h_k(info)$ and *ch* have the same number of bits, and that the number of bits in *iCVV* is equal to the number of *1-bits* in *ch*.

To verify that their implementation works with the functions *H, G, F*, the XOR operation satisfies *F1, F2*, so satisfies function *F*. And the *khi* which is generated from $h_k(info)$, here h_k is keyed hash function and these bits are indistinguishable from random to any party without knowledge of *info* and the banks secret key k. These bits are masked by the *iCVV* and as such are not learned by any party. Their implementation of function *G* satisfies the property *G1*.

They acknowledged that changing the architecture of a production system is difficult, particularly so within the payment industry. They proposed a similar but alternative protocol which avoids modifying currently running payment processing systems, by separating the verification of B from the processing of the payment. This could be done by deploying a proxy, capable of verifying B and translating charge requests into those of the current CC Protocols format. As a result, the NFC portion of the protocol enjoys the protections afforded by their Secure CC Protocol, while the payment processing systems continue to use the current CC Protocol.

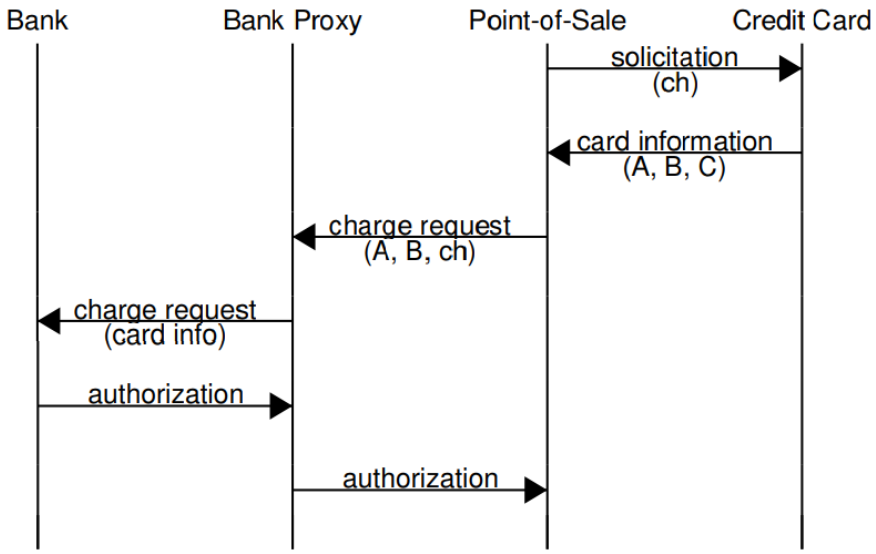

Figure 2.10: Separated Secure CC Protocol

This proxied protocol Figure 2.10 requires the creation of a new entity, the bank proxy, which is capable of looking up credit card information based on UUID, verifying *H(info, ch, iCVV)*, and translating charge requests of their Secure CC Protocol into charge requests as expected by the current CC Protocol. Due to this translation, the payment processing service which accepts or rejects payments can remain implemented exactly as is with no modifications whatsoever. The protocol as experienced by the Point-of-Sale and the credit card is identical to their Secure CC Protocol, and thus enjoys the same protections from the four classes of attacks. But this proxied protocol is also very expensive to implement as it needs to introduce a new element between bank and POS communication.

2.6.2 Jamming

SungTaek et al. [21] proposed a countermeasure of Relay Attack with jamming. They assumed that the attackers would communicate with each other via a different means of wireless communication other than NFC so that Jamming won't block the NFC communication. The assumption was that Jamming signal would be generated around the reader as could be seen in Fig 2.11 at the start of a NFC communication, so that Relay Communication would be blocked. And the jamming device can be connected to the system with reader and would be controlled by the reader. Proposed system has a series of communication procedures shown in Figure 2.12. Traditional proposals are mostly on detecting the attack but they proposed of blocking the attack.

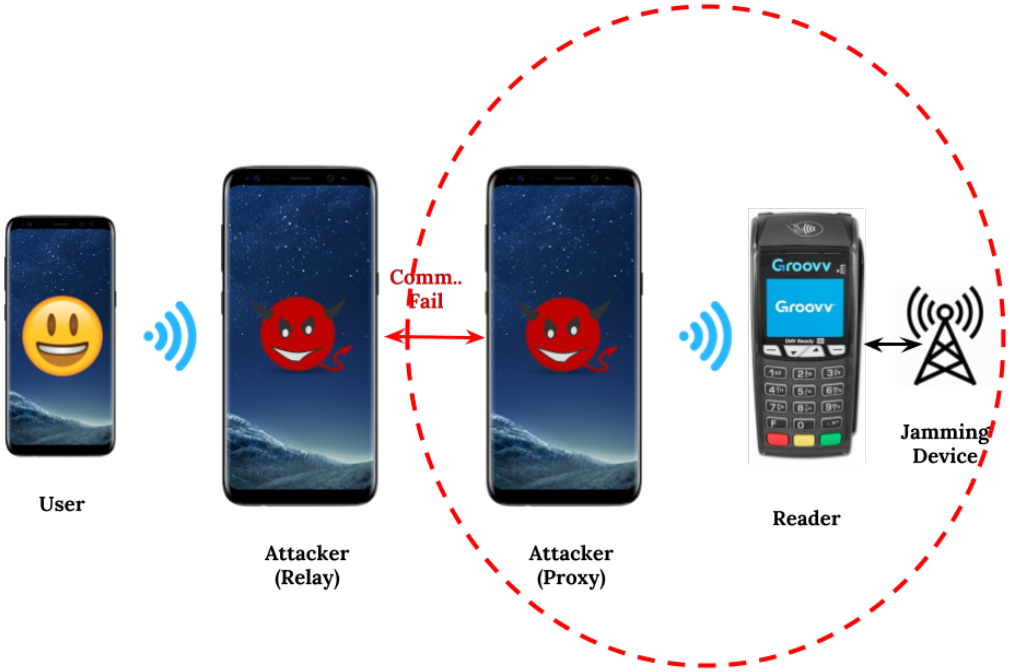

Figure 2.11: Proposed System Diagram

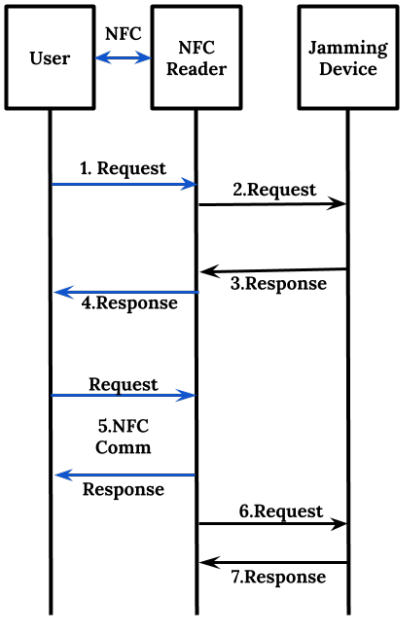

Figure 2.12: Series of communication procedures

As there was no implementation given on how the jamming device would be and the controlling of jamming device from reader is like proposing a modified reader which is able to send jamming signal, it would be very costly to change the current trends of payment industry.

2.6.3 Secure EMV Protocol

Tom et al. [22] proposed a secure protocol based on the delay time of Relay Attack. When Card needs to encrypt data before reply and Card doesn't need to do any computation before reply, there is a huge difference in the amount of delay a Relay Attack creates. The delay in Relay Attack during a Card reply when no computation needed is large enough to detect compared to when the card does computation before reply.As we can see in Figure 2.13 the current Visa payWave qVSDC Protocol.

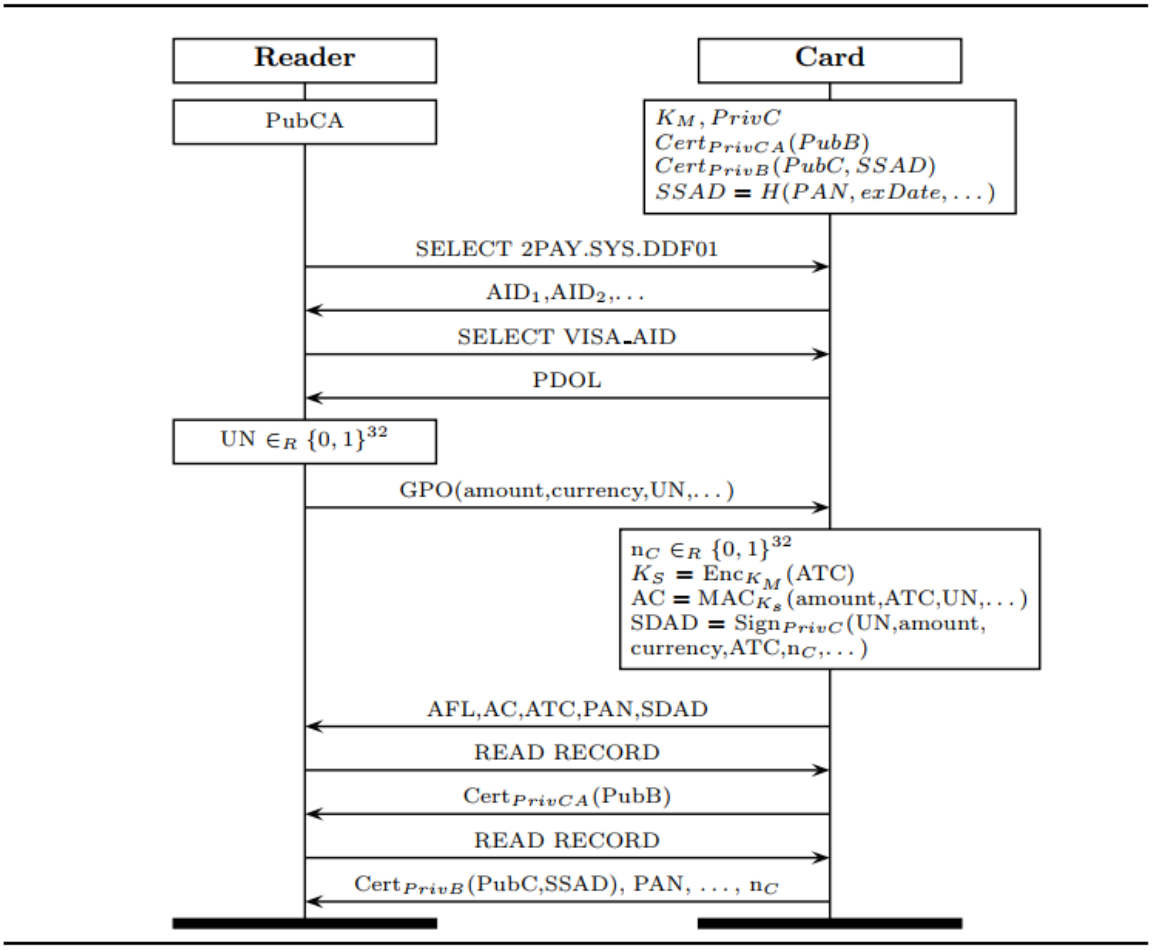

Figure 2.13: Visa's payWave qVSDC Protocol

The problem with using the small delay caused by the relay to detect attacks is that this cannot reliably be used for commands that carry out cryptographic computations or return data that can be cached. The time of the cryptographic computations will vary due to for instance, the cards hardware or placement in the field, and so cannot be reliably bound.

They proposed that the problem can be fixed by splitting the challenge and response command from the generation of the signed authentication and cryptogram. To do this they provided the readers nonce to the card with the GPO command (as in the payWave protocol), but they delay the generation of the signed authentication and cryptogram until the

reader issues the GENERATE AC command (as in the PayPass protocol).

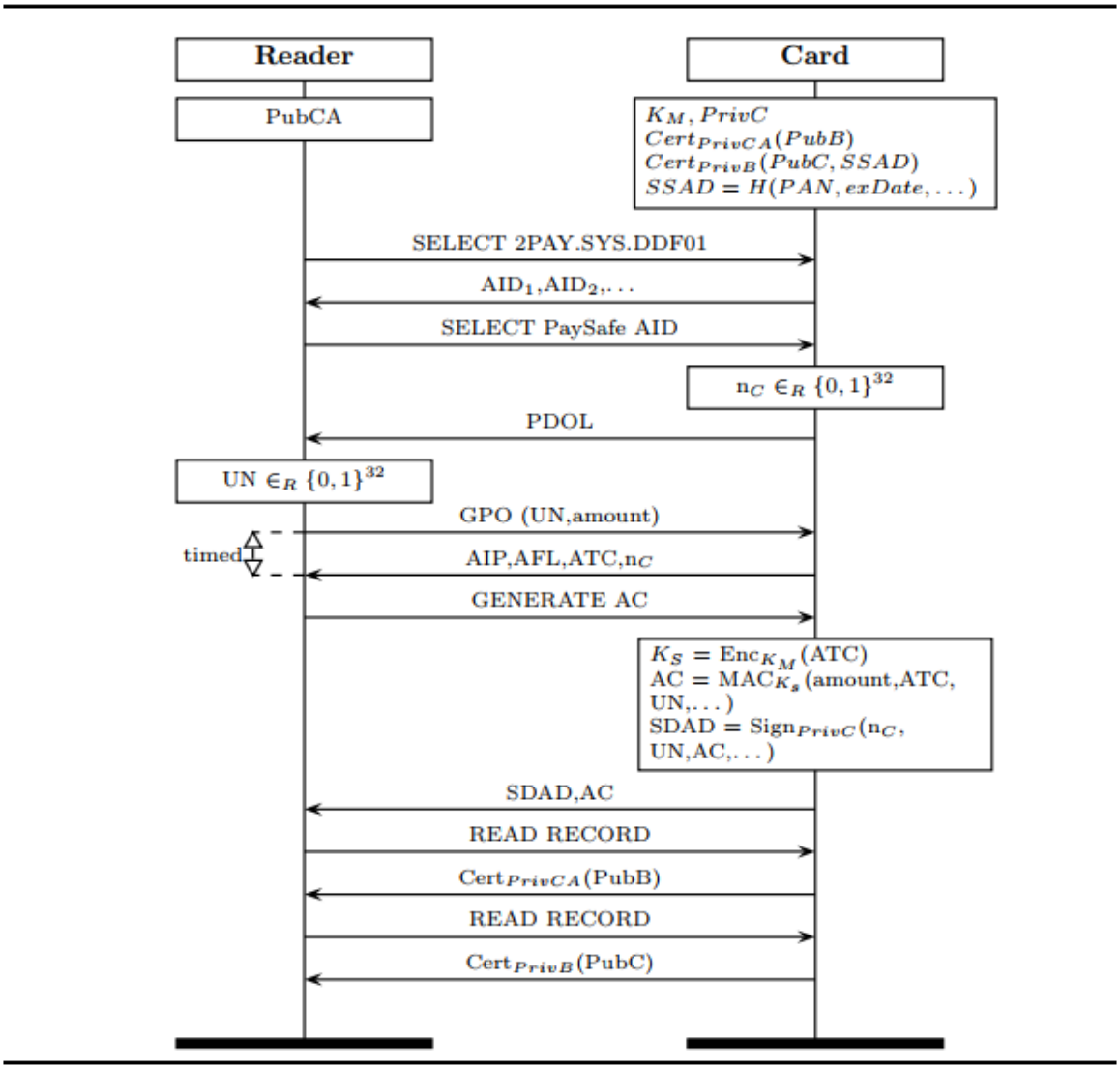

Figure 2.14: The protocol PaySafe which defends against simple relay attacks

They named their protocol ”PaySafe”, is shown in Figure 2.14. In reply to the GPO command, the reader would provide its own nonce that was generated at a previous step in the protocol (i.e. when receiving the SELECT command). As the reply to the GPO command now does not require any computation, it can be timed by the reader to detect relay attacks. To detect an attempt by an attacker to defeat the timing bound by generating their own nonce to replace the readers or cards nonce, both the reader's and the card's nonce are included in the signed data, as is the AC, so fixing VISAs problem of allowing an attacker to inject an invalid AC.

They used existing fields within EMV to exchange the nonces, namely the Unpredictable Number and the ICC Dynamic Number for the reader and card respectively. That way the cards were still EMV compliant and additionally, since both values are included in signed

data, no need to add an additional signature to the protocol.

Their proposed protocol would start in the same way as the existing EMV Contactless protocols with the selection of the payment application. After the application was selected, and before the card sends its request for data to the reader (the PDOL), the card would generate a nonce. This nonce was not sent to the reader at this point but just stored on the card.

The next step was a timed GPO command, that gives the card the data it needed and the reader's nonce. The card would immediately reply to this message with the stored nonce. As this didn't require any computation the card would reply quickly and without much variance in the time taken.

If this message was relayed, additional overhead would be introduced by the network communication between the phones and the communication between a phone and the genuine terminal. The exact timing would depend on the hardware used; with their hardware the reply to the PayPass GPO message took 36ms and the PaySafe message was slightly longer.

Therefore, they suggested a time out of 80ms, as being long enough to make sure all cards were accepted, but still quick enough to make sure the message couldnt be relayed using NFC phones.

To get the cryptogram and Signed Dynamic Application Data (SDAD) from the card, the reader would issue the GENERATE AC command. The card then computes the SDAD and the cryptogram and sends these to the reader. Attackers could get to this point in the protocol by sending their own nonce to either the card or the reader, so avoiding the need to relay both nonces and meeting the time restriction. To detect this both the readers and the cards nonces would be included in the signed data (SDAD).

The reader would check these and terminate the transaction if they didn't not match the nonces that were part of the timed challenge. After running this protocol the reader could be sure that fast nonce exchange took place with an entity that was not being relayed using mobile phones or USB NFC reader links, and, due to the SDAD, the reader could be sure that this entity was the bank card.

They also added that variations of this protocol were also possible, for instance the PDOL could be replaced with a CDOL in one of the records and the GENERATE AC message could be moved to the end, to make it much closer to the current PayPass protocol. It would also be possible to move the AIP,AFL and ATC data from the reply to the GPO into the reply to the SELECT AID, so making the reply to the GPO quicker and easier to time. However that would be a further deviation from the current EMV protocols.

As they proposed an EMV protocol which would need cheap hardware to implement and according to their experiment, the timing should be 80ms to detect a Relay Attack and Relay Attacks were successfully detected, It seems a successful protocol for detecting Relay

Attack. But as that mostly depends on the generation of Unpredictable Nonce but most of the time the nonce can be predictable. And the changes they proposed was small but it would affect both the reader and card. The payment industry is very large and adapting the changes would be a risk for the large organizations. So it's not yet feasible to adopt the protocol they proposed.

CHAPTER 3
RELAY ATTACK ON NFC APPLICATIONS

The *partial* goal of this thesis is to deliver a successful NFC relay attack system via off-stock market phones. The final goal is to study the relay attack by smart phones and propose methods to detect and prevent it. We use off-shelf Android smart phones as hardware and build applications on them. We install modified open source operating system lineage os on them. We also using Java reflection technology to expose related NFC emulation APIs from the operating system. Within the powerful handset, we are able to emulate a practical tag at the same time communicate the data we received from real reader back to the smart phone which is operating in read/write mode, where the tag is actually read/written according the transaction.

This chapter consists of information and perquisites of derive a practical attack on NFC applications. We first discuss about the basic of general relay attacks, and then talk a little about the requirement of NFC relay attack. We explore in-depth about how far we can achieve via the existing operating system and hardware, and give out an overall design of attacking system.

3.1 Man-in-the-middle attack (MITM) and Relay attack

The very beginning of implementing this practical attacking system is to understand the fundamental concept of relay attack and its similarity to Man-in-the-middle(MITM) [23] attack.

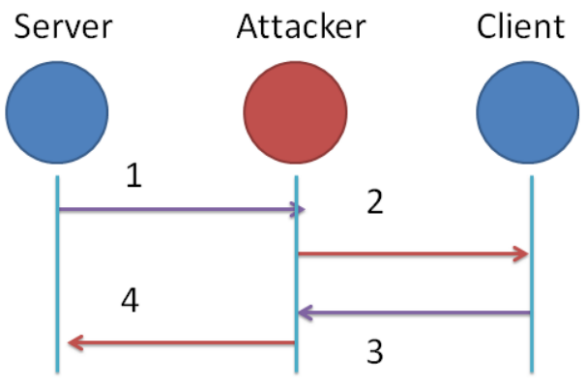

Figure 3.1: Man-in-the-middle attack workflow

Man-in-the-middle attack, is often refer to a kind of attack that that a third party is interfering with the normal traffic between two parties to gain information about the communication. The attacker hijacks the communication channel and impersonates as the normal users. The entire communication is interrupted and legitimate users at end points still believe they are talking to each other without a third party in the middle. The attacker must be able to achieve the following points to make a successful MITM attack:

- **Interrupt the whole communication between two legal parties :** This is achieved via hijacking communication, e.g. pretending to be the access point of Wi-Fi to hijack all internet traffic [24]. This is not easy to achieve in telecommunications.
- **Impersonate as legitimate users in the conversation :** For simplicity, if there is a server-client mode service running, MITM attacker shall be able to impersonate both as client and server.
- **Reproduce exchanging information :** This step is the most difficult part. Usually modern communication protocol(such as SSL/TLS) is encrypted and needs authentication steps (to both end-users) to generate session key to encrypt the channel. Or the session key is generated based on a shared secret for both parties. Without the session key, an attacker can do nothing with the actual content in fly. Nonetheless he inserts new, valid traffic in to communication, e.g. An attacker shall be able to have the symmetric encryption key to encrypt and decrypt traffic. Otherwise attacker can't access the communication channel.

As we can see from the illustration Figure 3.1, the attacker will receive communication from server at step one and try to decrypt it and then encrypt it again to send out at step two. The same thing would happen if attacker receives information from client on step three. However, if the attack fails to obtain the session key for original message encryption, he cannot fully understand traffic and reproduce correct message. In our very specific scenario NFC applications, unfortunately the communication can be encrypted by DES/AES/3DES. Some communications can be plain text or MACed messages thus attacker still can sniff them.

Here comes the relay attack, a close variant of MITM attack. In a "reader - attacker - card" system, an attacker can forward information without recalculate it. It is just like a router, forwarding IP packages without understanding the payload of it. But the router still can dump the traffic and study the traffic time pattern. Such kind of attacks is easy to implement as long as we can emulate the parties in the communication. Fortunately in our NFC cases, although the traffic has an option to be encrypted, it still lacks of a detecting system to find out the existence of possible relay attackers in the middle. As operating in a close distance

as a design and pattern, NFC is totally vulnerable to relay attacks. The Figure 3.2 is an example of design a two-phone solution to derive such kind of relay attack.

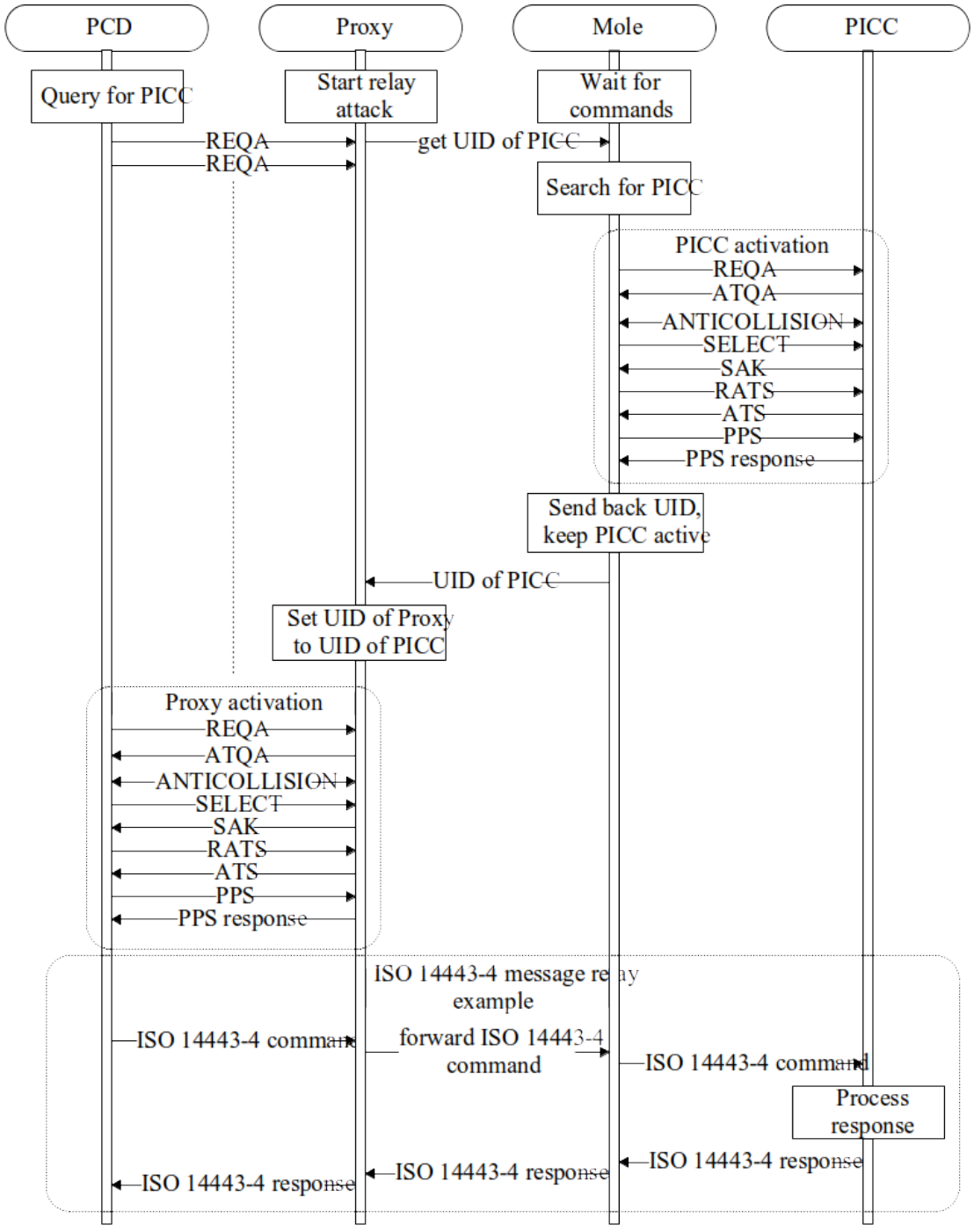

Figure 3.2: General Concept of a Relay Attack

Here a cell phone is used as a fake card (Proxy) to communicate with the real reader (PCD), using card emulation technology to emulate a smart card. Another cell phone is acting as a

fake reader (Mole) to communicate with the real card (PICC). The two cell phones (Porxy and Mole) are linked with a wireless channel (Wi-Fi or Bluetooth). Once the real reader initiates transactions, all the request messages are forwarded through wireless channel from Proxy to PICC. PICC conducts real transaction with smart card using the request messages. The transaction responses from real card are transmitting back to the real reader via the same channel. This relay attack makes the relay attacker transparent to both NFC reader and real card.

3.2 Requirements of Smart Phone NFC Relay Attack

This section we discuss the basic requirements if we want to make a NFC relay attack using off-shelf Android smart phones.

3.2.1 Phone-to-Phone Communication Channel

As we have discussed above before, NFC relay attacks have to solve the problem of distance in the first place. Usually the card is read and written at a service point such as POS machine. If an attacker have a wired communication cable between your attacking devices, it not only limited attacking range but also make attacker more suspicious in public.

Fortunately, the modern Android smart phones are equipped with enough communication ability such as Wi-Fi or Bluetooth. This serves as a best invisible proxy channel. NFC transactions limit the length of payload by 253 bytes so the total on-fly traffic would be less than 300 bytes [25]. Regardless of means of transmission, this payload will be quickly transmitted over Wi-Fi/Bluetooth. The design of communication results in three styles of link between our attacking devices:

- **Wi-Fi directly link:** In this mode one cell phone is acting as a router and the other as a regular Wi-Fi client. Information is passed without going through a third router.
- **Bluetooth directly link:** In this mode two cell phones are connected via Bluetooth in pairs. Information is also passed without a third party.
- **Mobile Broad Band links:** In this mode two cell phones will both sends information to a remote server on the internet acting as an exchange server. Communication is indirect but has the most flexibility. Also this channel is provided by a reliable telecommunication company.

The obvious advantage of first two methods is they are controllable in terms of time delay. Communication time is highly dependent on the Wi-Fi and Bluetooth technology itself and

operating system stacks on each phone. However, the shortcoming is the devices must be configured and paired before they can be used. Also the distance of attack is limited by the max Wi-Fi or Bluetooth transmission range between two devices.

The third option is less reliable. As we rely on a third party telecommunication service, delay time is uncertain. Also it requires a reflection server operating constantly on the remote, this will increase the delay in the communication.

3.2.2 NFC Read/Write, Emulation Mode

Nowadays a smart phone is as a powerful personal computing base.Popular smart phones are equipped with NFC chips and the APIs for those smart phone platform enable the programmer to easily implement their logic. So these smart phones become more suitable for conducting NFC relay attack. Take Android smart phones as an example, although different vendors has different models, most of them have NFC chips and a unified API interface exposed by Android operating system. Read/Write of NFC bus cards is no longer bound to specific top-up machine as a user can easily check his bus card balance on special designed app by tapping his card on the back of Android smart phone. Programmers area able to write applications which emulates a NFC card (Host Card Emulation).

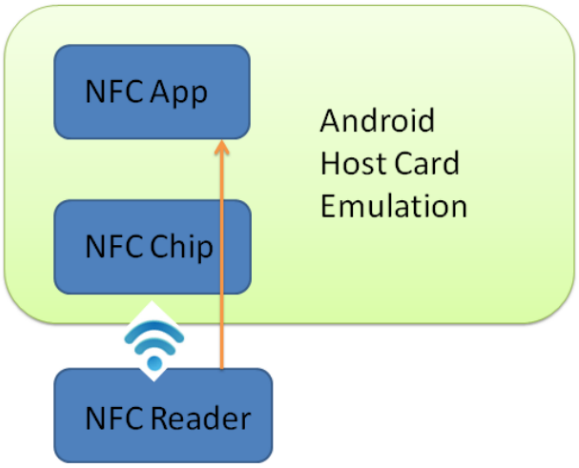

Figure 3.3: Host Card Emulation of NFC on Android

As the Figure 3.3 shows the design of an Android native application to use host card emulation (HCE) feature is highly controlled by Google Android operating system. It only exposes the interface of ISO 7816 application level APIs.

When the HCE device is approaching a reader, lineage operating system will generates an intent and invoke the NFC app designed by us. This app emulates the interface as a DESFire EV1 card. Requests from reader are sent directly from Android operating system to NFC app. These requests are then transmitted to another smart phone which is interacting with a

real card. While functionality is described simply in design, how to let it cooperate with the communication channel is difficult during implementation. We have to carefully optimize the application structure to accelerate app logic speed and achieve low latency of our relay attack. The transmission of NFC messages to Wi-Fi/Bluetooth channel shall be efficient and error tolerated. We also have to make the program code separate and components de-coupled so maintenance of code is easier.

3.2.3 Block Communication between Reader and Card

The nature of NFC is a contactless transmission protocol in short range only by 1-10cm, we can achieve blocking communication by attaching attackers Mole phone directly on the NFC card to avoid other devices reading it. Furthermore, put the NFC card away from reader and attach another Proxy phone to its reading surface. By the design of Android operating system, only one intent is triggered if we are doing NFC transaction. We design our application to be triggered and keeps foreground running during the attack. This forces the communication between the reader and card to go through our attack devices only.

3.3 Solution Design Components

In this section we will cover the general choices of components to build up feasible relay attack software architecture on Mole phone and Proxy phone. These software components form Android apps to run on the phones. Here we introduce the components on high level and justify our choices.

- **Wi-Fi and Bluetooth direct links:** As we have discussed, using a third router or mobile broad band would significantly increase the potential delay time in transmitting the data.
- **Lineage os** enables us to manipulate on ISO 14443-4 level and we can observe raw APDUs from there. Android 7.1 only allows programmer to visit ISO 7816 level, which is a level higher than ISO 14443-4.
- **Control communication from ISO 14443-4 transmission level** We leave the lower level to the operating system to deal with. ISO 14443-3 anti-collision is performed by under-lying operating system. Our app program logic starts to forward actual APDUs on the ISO 14443-4 level.
- **Device Role Divide** One smart phone is called Proxy, it is in card emulation mode and expose its interface as Mifare Classic 1K to the real reader. Another smart phone

is called Mole, it is in read/write mode and gets transaction instructions from the Proxy phone over Wi-Fi or Bluetooth. Mole phone receives the requests and deals real transactions with a Mifare Classic 1K Card. No third device is required.

The above first point and last point is easy to understand in our scenario. Only two phones are involved and we choose a wireless channel of Wi-fi/Bluetooth to link them up. But the second and third are related to physical constrains we have. Lineage operating system is a modification of Android open source project. It already introduced API level host card emulation although quite primitive, limited only to ISO 14443-4 level (no above layer ISO 7816 support). The development progress is also painful. It also requires the programmer to use Java reflection technique to wrap around the corresponding APIs(these APIs are not part of the Android Development SDK release). However, the raw APIs on ISO 14443-4 level gives us great chance to truly look at the content of original NFC requests/responses in bytes. Also Lineage is a tested operating system version which is compatible with our hardware existing in laboratory HTC ONE M9 smart phones.

In the third decision we decided to forward NFC commands on ISO 14443-4 level. It is because the lower level ISO 14443-3 anti-collision is not exposed to client application in Android even in modified Android. And since lower level ISO 14443-3 level provides anti-collision and activation at the beginning of the conversation phase, we have no choice but rely on underlying operating system to handle it then take the control over once it is done. From ISO 14443-4 level, our app logic begins the monitoring/forwarding on real NFC transaction commands.

3.4 Prevention Proposal

Here we present our prevention proposal which is based on the time delay of Relay Attack. Here we can see in Figure 3.4 a normal NFC communication and the travel time it takes. If we add a clock like device or a counter in reader, which measure the time for a reply from Card, then the round-trip time measurement can be achieved.

At normal communication, POS terminal determines the time as $Tpos = T$

Here T = *travel time from POS to Card (TT) + Process Time of Card + travel time from Card to POS (TT)*

We assume in our proposal that, Card also has a built-in clock or counter which measures the process time for a reply and adds it the packet it sends.

Card determines the time as $Tcard = t$

Here t = *Process time of Data + Adding data to Packet*

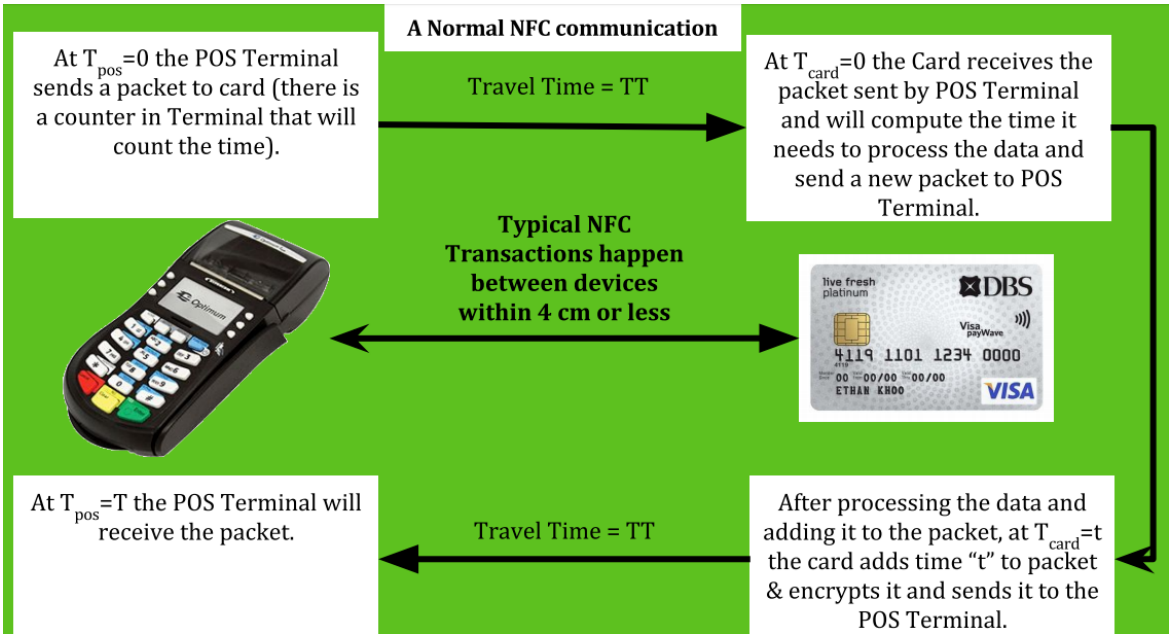

Figure 3.4: A Normal NFC Communication

But there is some additional time Card needs to add the time (t) to packet and encrypting the packet.

So, *Process Time of Card = Process time of Data + Adding data to Packet + add the time (t) to packet + encrypting the packet*

But the time to add time(t) to packet is almost constant on every time the Card sends the packet.

And the encryption time may vary on the size of the packet.

But the packet also has a constant size. So, Encryption time will also be a constant.

Actually the whole process time of Card is almost constant for a Normal NFC communication Figure 3.4 with two entities .

But from card to card the process time will vary and If its not card rather Its a smart phone than the process time will vary from the cards.

The reason we need the process time is - we need to calculated the travel time.

In normal communication,

Travel time or Round-trip time = travel time from POS to Card (TT) + travel time from Card to POS (TT) = 2TT

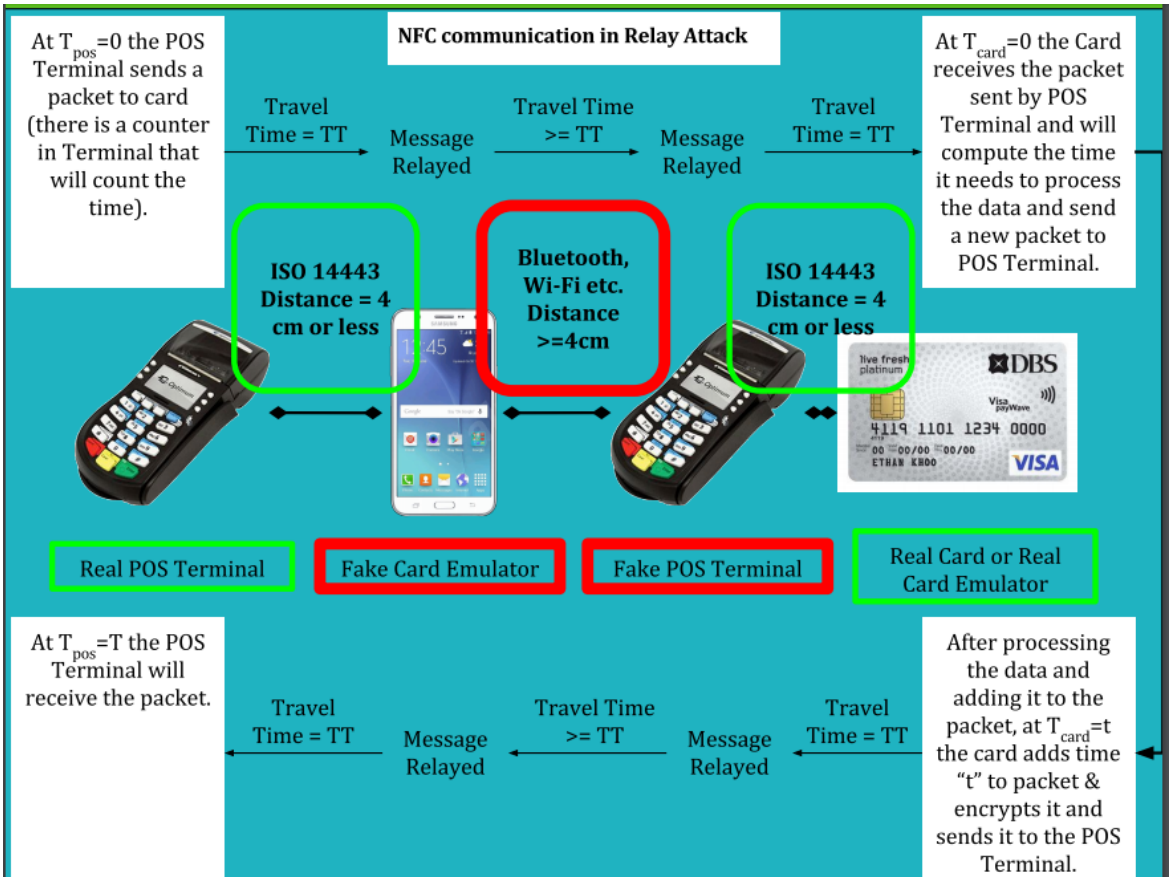

Figure 3.5: A Relay Attack Scenario

And here we can see the Relay Attack scenario in Figure 3.5. As there is the presence of Fake Card Emulator and Fake POS Terminal, the travel time will be longer than a Normal NFC communication.

So according to our proposal, the Reader will measure

Travel Time

= travel time from Real POS to Fake Card (TT) + travel time from Fake Card to Fake POS (>= TT) + travel time from fake POS to Real Card (TT) + travel time from Real Card to Fake POS (= TT) + travel time from fake POS to Fake Card (>= TT) + travel time from Fake Card to Real POS (=TT)

= 4TT + 2(travel time between fake POS and fake Card)

So, when the POS gets the Card Data Processing time(t), then It will add some extra time(t_{extra}) to compensate the (Add time(t) to packet + Encryption time).

And subtract it from the Reader calculated time Total travel time = T - (t + t_{extra})

In normal communication, transfer rate = 106kbps, 212kbps, 424kbps.

For the real POS & real Card distancing 4cm or less, Travel time (TT) is almost same

because it always uses ISO-14443.

But between fake POS & fake Card, the connection methods can be Wi-Fi or Bluetooth or any other technology but not ISO-14443. So it will definitely take longer time then NFC communication.

So the POS will easily determine the difference in Total travel time difference and detect Relay Attack.

CHAPTER 4
OUR APPROACH TO ATTACK

This chapter covers the procedure of our approach to enable a relay attack on NFC smart cards. We tried to implement a prototype as we discussed in Section 3.3. The whole chapter is divided in three parts.

First we discuss which development tools we have been able to choose. We needed one mobile phone and we modified an app for our specific NFC architecture so that we can perform basic NFC operations.

And then we focus on how we arranged the setup, so that we can show the general work flow of the app.

Then we discuss the important codes and functions of the app we modified. NFC basic functions such as Reading and Writing on NFC tags.

4.1 Development Tools and Dependencies

For the basic developer machine, a Windows 10 is with Java 8 is required (Used to develop NFC apps on HTC One M9 smart phones). In addition, there would be two already rooted HTC One M9 smart phones with NFC features enabled. These two phones are installed with Viper One M9 6.2.0 operating system which is similar to Lineage OS. Although it is not the newest one in the market, it features enough battery power to do the experiment and is available in our country. These requirements are described in table Figure 4.1.

Model	NFC Controller	Bluetooth/Wi-Fi	OS
HTC One M9	NXP 47803	Supported	Viper One M9 6.2

Figure 4.1: HTC One M9 with Viper OS

We had to both re-install TWRP and install Viper One M9 os. The modified system exposes NFC host card emulation feature which cannot be accessed in built-in Stock ROM.

We needed a Proximity Integrated Circuit Card (PICC) which is an NFC Tag or card. For payment transaction, we needed Mifare Desfire EV1 [26] but isn't available in our country.

So we used Mifare Classic EV1 1K [27] which is basically used to store info. Although it has support upto ISO-14443 Part 3 that means It can't be used as credit card and will not support payment transactions as Payment needs ISO-14443 part-4 support.

Figure 4.2: Mifare Classic 1K

Our NFC controller NXP 47803 (can be seen in Figure 4.3 with red border) is integrated into HTC One M9 phone for NFC purpose. But it hasn't been implemented yet on any app that is open source, we felt we need to build an app from scratch which will be able to do Relay Attack. Because no previous app both open and closed source has support for this NFC Controller. So we needed thorough technical details on apps that were published for NFC. We found very small amount of technical details available online. Finally we found an open source app *Mifare Classic Tool* [28] which we can use to do a Relay Attack. The app has been open source since 2013.

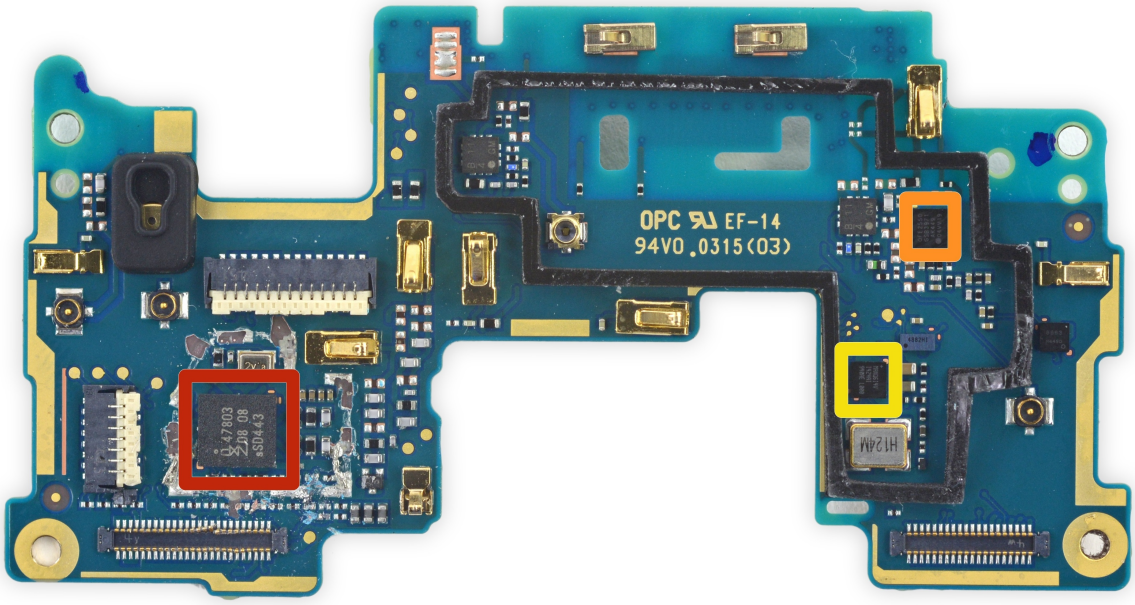

Figure 4.3: NXP 47803

4.2 Work Flow

For the basic setup configuration, we needed one Phone and One Card. As we have available HTC One M9 phone and Mifare Classic 1k card, we went with that.

First, we installed the app we rebuilt Mifare Classic Tool for our specific NFC controller NXP 47803 on HTC One M9 phone.

Next, we turned on the NFC of our phone.

Next, we approached our HTC phone towards a Mifare Classic 1k card.

Next, when we reached as close as 10cm or less, a pop up will tell us that it has detected an NFC tag.

Next, now we can use the app to read the tag. We first need to choose the key files which contain the keys for the tag and select start mapping button on the right, to start mapping the tag.

So we will then see the memory info of the tag as shown in Figure 4.4

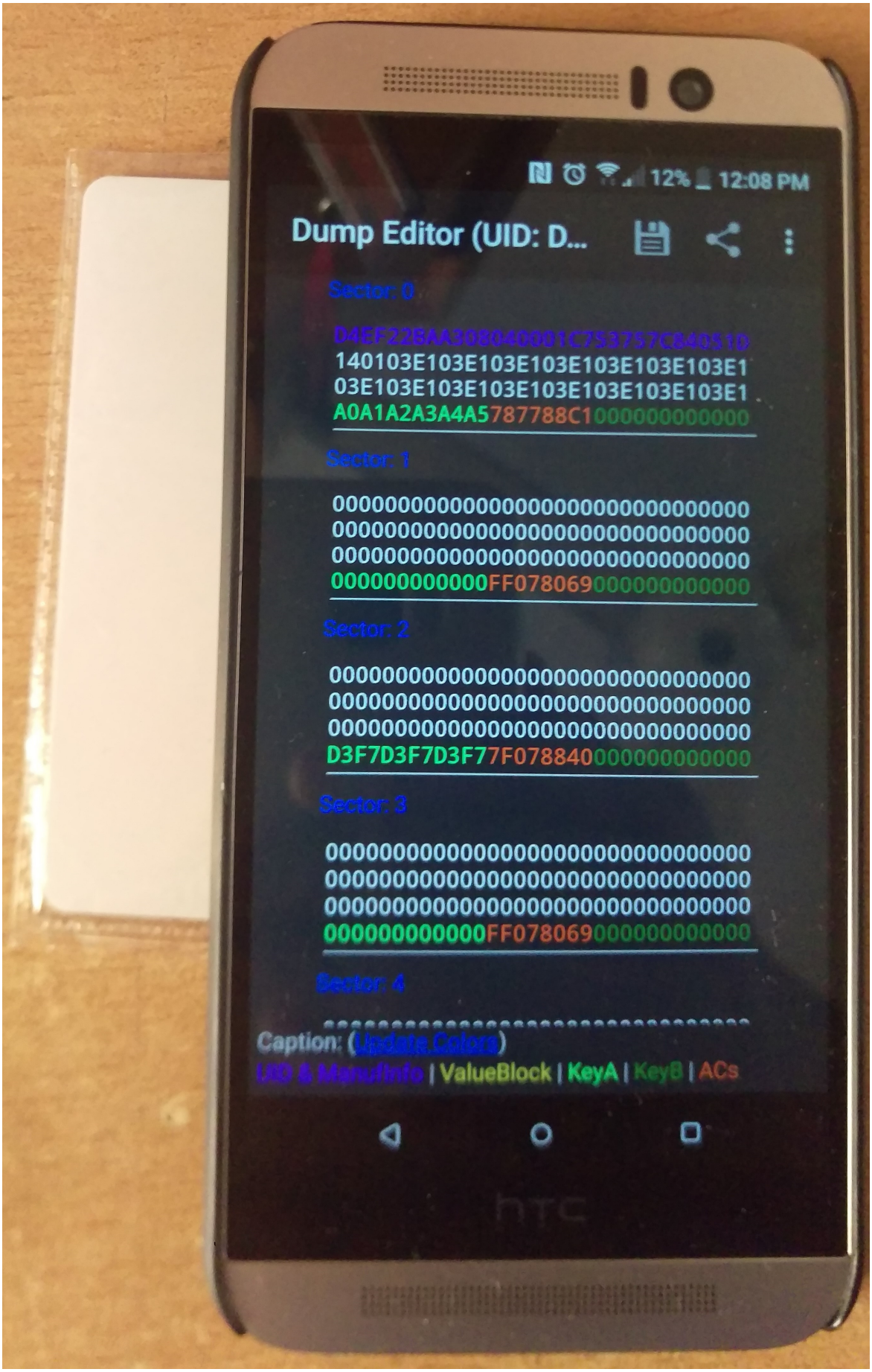

Figure 4.4: Dumped Data from Mifare Classic 1k

4.3 Read Tag Memory

Our MiFare reading app reads the NFC Tag and dumps the data from memory. Mifare app needs certain keys to read the memory which are already stored on the app.

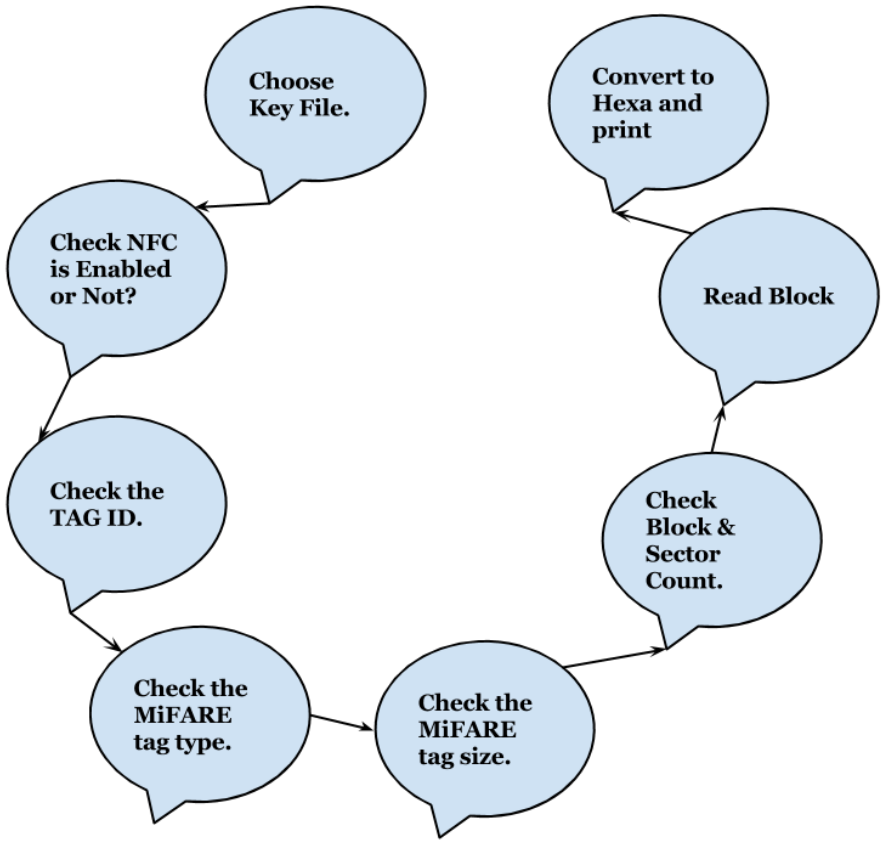

Figure 4.5: Read Tag Process Sequence

As we can see the from the Figure 4.5 that first we need to select key files which will enable us to see the content on the Mifare Card [29]. Then with the key, we will read the blocks of the MiFARE Card.

A Mifare Classic 1k tag contains 16 sectors. Each of these sectors has 3 blocks of data storage and 1 block for storing the secret access keys and access controls. Each block contains 16 bytes of data. Before reading a sector, the reader must authenticate to the tag with a secret access key. Each sector has two keys: Key A and Key B Each of the 16 sectors can define its own access right and wich key is needed for a particular action. As an example you can define to use Key A for reading the block and Key B for writing to it. Sector 0 Block 0 also contains a non changeable UID (the tags unique ID) and some manufacturer data. This section is only writeable on some special chinese tags.

Here 4.6 is a Mifare Memory Layout.

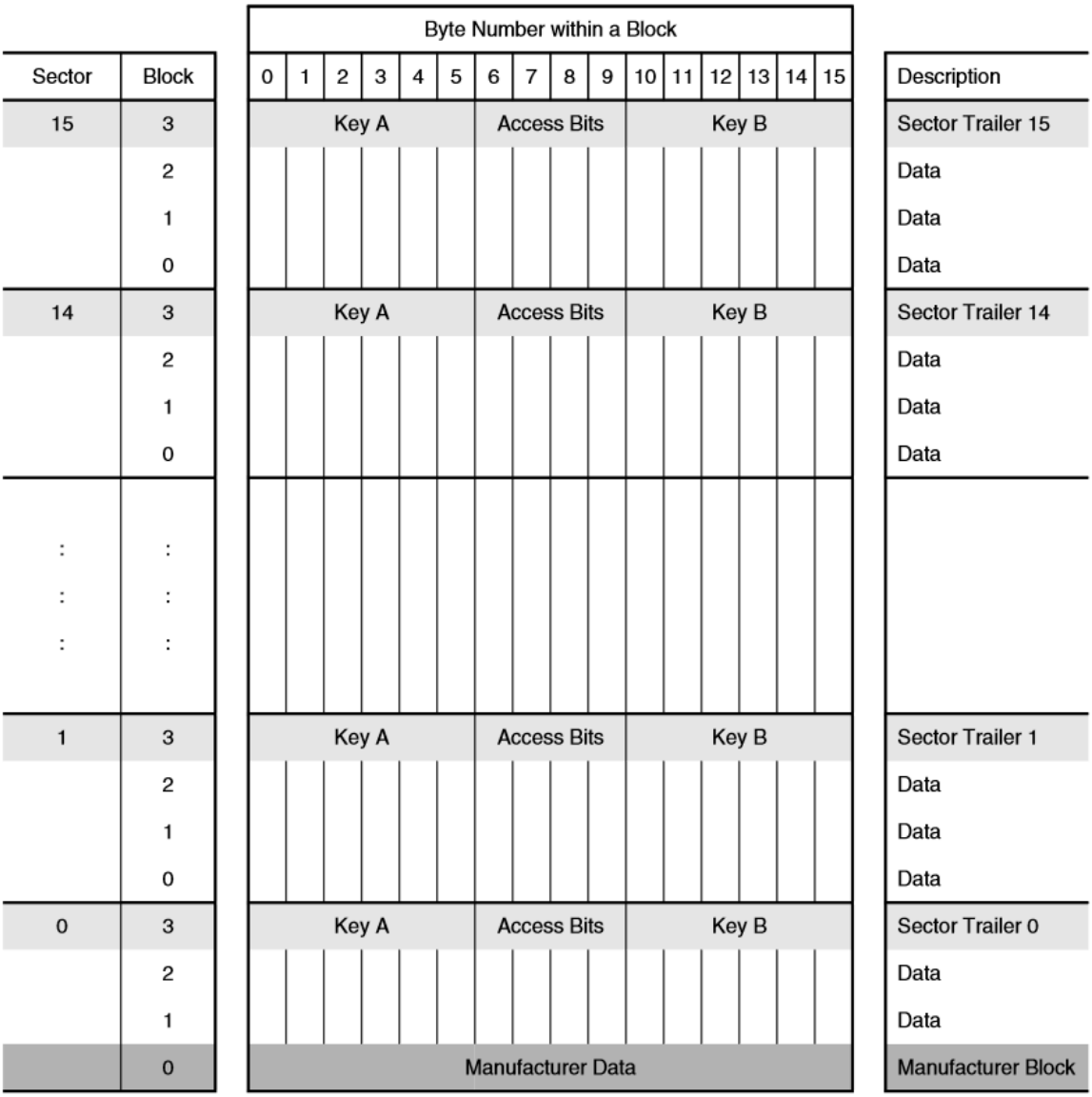

Figure 4.6: MiFare Memory Layout

Next our app will check if the Phone is NFC supported. And if the phone is NFC supported, then NFC must be enabled which will checked by our App.

Next we will check the TAG ID and TAG type. After knowing the appropriate TAG type, we will check the TAG size, Block and Sector Count.

According to Block and Sector Count, the app will read the strings and convert to Hexadecimal and dump them. That is how the Read Tag process in done.

CHAPTER 5
OUTCOME

Our App has successfully able to read the memory of a Mifare Classic 1k card. It supports reading most of the Mifare classic tags such as 1k, 2k, 4k etc. As measuring the time, we need a tool called Proxmark3 [30] which we were unable to buy.

5.1 Limitations

We faced many kinds of problems during our thesis. If we had sufficient equipment, we could have proved the prevention with proper experiments. The problems we faced are following :

- **NFC supported POS Terminal** isn't available in Bangladesh. For real time attack time measurements, payment time on Relay Attack must be measured.
- **Proper NFC chip** In our HTC phone, pre-installed NFC chip wasn't compatible with most relay attack apps. So we approached to build one.
- **Other Equipment** As we didn't have proper equipment available such as, Proxmark3, a tool which is popular among NFC and researchers, MiFare Desfire EV1 card, which can only be order by large companies mostly Banks and other attack tools.

5.2 Future Work

Our thesis topic "NFC" is quite a new one and currently there aren't many people researching on this topic. But soon there will be one day when NFC enabled phones will be on everyone's hand and contactless payments will become a norm. And NFC is quite a hard topic to grasp as there are not much technical details online and most researches are on specific hardware not generic which prevented us from going further. But we will continue our research on this topic and gradually we will be able to collect all the tools we need for successful Relay Attack and prevention. On the prevention attempts, as most prevention proposals including ours is kind of like changing the whole payment industry. So most of the proposals are still not implemented. If proposals have very little risk to implement and

no side affects, only then a proposal can be successful. As we grow deeper in understanding, we will try to implement the following four topics on NFC - (1) A Successful Relay Attack on NFC payments, (2) Other vulnerabilities of NFC features, (3) NFC data transfer speed is less than lower then Bluetooth and Wi-Fi, although NFC has the devices in the closest distance, So we will try to increase the NFC data transfer speed which is also mentioned in ISO-14443, (4) Adopting NFC on Laptops, Desktops for secure authentication and data transfer features.

CONCLUSION

Most companies like Apple Pay, PayPal or MasterCard PayPass use NFC for payment. In many countries most popular banks are giving NFC supported credit cards. And the use of NFC TAGs in Identification, Authentication and Access has been going over the expectation. Although NFC hasn't been yet adopted in Bangladesh, we hope very soon NFC will touch different sectors of our Country easing many aspects of our life. With the growth of NFC usage, so does the growth of its vulnerabilities increase. As a new technology, not many researches are going on NFC. NFC is considered as a secure method for payment except for eavesdropping, relay attack which can't be prevented entirely. But if users stay alert, it's possible to prevent attacks most of the time. As researches going on and we will relentlessly add our effort on the research of a safer wireless world, we hope to make NFC a secure payment method.

Bibliography

[1] S. C. Alliance, "Proximity mobile payments: Leveraging nfc and the contactless financial payments infrastructure," *Smart Card Alliance*, 2007.

[2] A. W. Ziv Kfir, "Picking virtual pockets using relay attacks on contactless smartcard," *IEEE*, pp. 47–58, 2005.

[3] K. B. Ernst Haselsteiner, "Security in near field communication (nfc)," *In Workshop on RFID security*, p. 1214, 2006.

[4] C. P. David Oswald, "Breaking mifare desfire mf3icd40: Power analysis and templates in the real world," *In Cryptographic Hardware and Embedded SystemsCHES*, pp. 207–222, 2011.

[5] P. K. V. Jarkko Sevanto, Petri Vesikivi, "Phone with secure element and critical data," *US Patent 7,694,331*, 2010.

[6] C. K. J. S. Gerald Madlmayr, Josef Langer, "Nfc devices: Security and privacy," *IEEE*, pp. 642–647, 2008.

[7] G. P. Hancke, "A practical relay attack on iso 14443 proximity cards," *University of Cambridge Computer Laboratory*, pp. 1–13, 2005.

[8] C. Mulliner, "Vulnerability analysis and attacks on nfc-enabled mobile phones," *IEEE*, pp. 695–700, 2009.

[9] F. P. Erina Ferro, "Bluetooth and wi-fi wireless protocols: a survey and a comparison," *IEEE*, vol. 12, pp. 12–16, 2005.

[10] Wikipedia, "Iso/iec 14443," 2017.

[11] I. J. S. 17, "Iso/iec 7816-4:2013 preview identification cards – integrated circuit cards – part 4: Organization, security and commands for interchange," 2013.

[12] I. J. S. 17, "Iso/iec 14443-4:2016 preview identification cards – contactless integrated circuit cards – proximity cards – part 4: Transmission protocol," 2016.

[13] I. J. S. 17, "Iso/iec 14443-3:2016 identification cards – contactless integrated circuit cards – proximity cards – part 3: Initialization and anticollision," 2016.

[14] I. J. S. 17, "Iso/iec 14443-2:2016 identification cards – contactless integrated circuit cards – proximity cards – part 2: Radio frequency power and signal interface," 2016.

[15] I. J. S. 17, "Iso/iec 14443-1:2016 identification cards – contactless integrated circuit cards – proximity cards – part 1: Physical characteristics," 2016.

[16] VISA, "Chip services simplifying implementation," -.

[17] V. Lorenzo, "Looking inside nfc security: Eavesdropping attack," 2014.

[18] T. W. C. B. S. W. Thomas P. Diakos, Johann A. Bria, "Eavesdropping near field contactless payments: A quantitative analysis," 2014.

[19] M. M. A. Allah, "Strengths and weaknesses of near field communication (nfc) technology," *Global Journal of Computer Science and Technology*, vol. 11, pp. 51–56, 2011.

[20] L. Q. Oliver Jensen, Mohamed Gouda, "A secure credit card protocol over nfc," *ICDCN*, 2016.

[21] T. K. J. K. M. H. SungTaek Oh, Taekyun Doo, "Countermeasure of nfc relay attack with jamming," *CEWIT*, 2015.

[22] J. d. R. J. v. d. B. M. T. Tom Chothia, Flavio D. Garcia, "Relay cost bounding for contactless emv payments," *FC*, 2015.

[23] V. L. Mauro Conti, Nicola Dragoni, "A survey of man in the middle attacks," *IEEE Communications Surveys Tutorials*, vol. 18, pp. 2027–2051, 2016.

[24] A. L. Marco Domenico Aime, Giogrio Calandriello, "Dependability in wireless networks: can we rely on wifi? security & privacy," *IEEE*, 2007.

[25] NXP, "An10833 mifare type identification procedure," 2012.

[26] NXP, "Mifare desfire ev1 contactless multi-application ic," 2013.

[27] NXP, "Mifare classic ev1 1k," 2011.

[28] ikarus23, "Mifare classic tool," 2013.

[29] FIREFART, "How to crack mifare classic cards," 2015.

[30] RyscCorp, "Proxmark3 kit," 2015.