Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

School of Computer Science and Statistics

An investigation of possible attacks on the MIFARE DESFire EV1 smartcard used in public transportation

Rory Flynn

July 24, 2019

A Final Year Project submitted in partial fulfillment
of the requirements for the degree of
B.A.(Mod.) Computer Science

Supervisor: Dr. Stephen Farrell

Declaration

I hereby declare that this project is entirely my own work and that it has not been submitted as an exercise for a degree at this or any other university.

I have read and I understand the plagiarism provisions in the General Regulations of the University Calendar for the current year, found at `http://www.tcd.ie/calendar`.

I have also completed the Online Tutorial on avoiding plagiarism 'Ready Steady Write', located at `http://tcd-ie.libguides.com/plagiarism/ready-steady-write`.

Signed: _____ Date: _____

Abstract

NXP's MIFARE range of smartcards are used as electronic tickets in transportation systems worldwide. Attacks that result in recovery of the master keys for a card (thus allowing an attacker to manipulate the data on the card) have been published for both the MIFARE Classic and MIFARE DESFire cards. However to our knowledge, no practical attacks have been published against the newer MIFARE DESFire EV1 card to date.

In this report, we focus on finding attacks against the DESFire EV1 application protocol itself. We create a simulated public transportation application for the DESFire EV1 and observe the raw traffic using a Proxmark3 RFID tool. Resulting from our observations, we outline 3 attacks that would allow an attacker to manipulate our simulated public transportation application.

We demonstrate a proof of concept for one of our attacks by developing a relay device that allows an attacker to modify commands and responses as they pass between a genuine card and reader. Using this device we demonstrate that an attacker can increase the balance on the card without permission from the system owner.

We critically discuss the attacks found, and plan to disclose our findings to NXP's Product Security Incident Response Team (PSIRT).

Acknowledgements

Thank you to my supervisor Dr. Stephen Farrell, for his sage advice and guidance over the past year.

To my family, who have supported me throughout my education - thank you.

To my friends in Computer Science, for making the last four years all the more special - thank you.

Special thanks are owed to Conal Cosgrove, who had the misfortune of discovering that an Irish Rail exit barrier will automatically charge you the maximum fare if you forget to tag on at the beginning of your journey! Your sacrifice was not in vain, and informs part of the attack detailed in Chapter 7.

Contents

List of Figures

List of Tables

Listings

1 Introduction

Report Structure:

In **Chapter 2**, a general background to RFID is provided, detailing its use in public transport, the MIFARE range of smartcards and existing security research on the MIFARE Classic, DESFire and DESFire EV1.

In **Chapter 3**, the research methodology is outlined, stating the aims of the project and providing background on the hardware and software utilised during the project.

In **Chapter 4**, the functionality of the MIFARE DESFire EV1, its file types, authentication procedures and encryption modes are detailed.

In **Chapter 5**, the development of an ISO-14443 RFID relay device is demonstrated. The difficulties encountered during this development process are also discussed.

In **Chapter 6**, the first attack found is presented. Observations regarding functionality of the Credit Command which enable the attack are presented, as are results of a Proof of Concept implementation of the attack.

In **Chapter 7**, the second attack found is presented. Observations regarding the lack of authentication for card responses to a WriteData command are made, and an example attack scenario is provided.

In **Chapter 8**, the third attack found is presented. Observations are made about the functionality of the anti-tearing mechanism and Commit command, and an example attack scenario is provided.

In **Chapter 9**, the impact of the three attacks is critically discussed, and possible mitigations are suggested.

In **Chapter 10**, the work carried out in the project is reflected on, and future work is suggested.

2 Background

This chapter provides a background on RFID and its use in public transport systems. It introduces the MIFARE smartcard range and details previous security research on the MIFARE Classic, DESFire and DESFire EV1.

2.1 RFID

Radio Frequency Identification (RFID) tags are traditionally used as a means of identifying objects or individuals for tracking purposes. The use-cases for RFID have expanded widely over the years to include access control, national identity documents and electronic payments.

2.1.1 Types of RFID

There are two general categories of RFID tags: active and passive. Active tags contain a battery or other power source which is used to transmit information which can then be picked up by a reader. In contrast, passive tags contain no internal power source, and must be powered entirely by the magnetic field from an external reader. This report will deal exclusively with passive tags.

Passive tags are further divided into High-Frequency (HF) and Low-Frequency (LF) tags, most commonly operating at 13.56 MHz and 125 KHz respectively.

High-Frequency cards are sometimes referred to as Near-Field Communication cards (NFC), as they operate in the near-field region, up to a maximum of 20 cm from a reader. Cards are powered by the reader's magnetic field through Faraday's Principle of Magnetic Induction, as shown in Figure 2.1. Data in the form of commands are transmitted to a card by varying the reader's magnetic field. The card sends responses by performing "load modulation" which can be detected by a reader as a change in current in the coil generating the magnetic field [2].

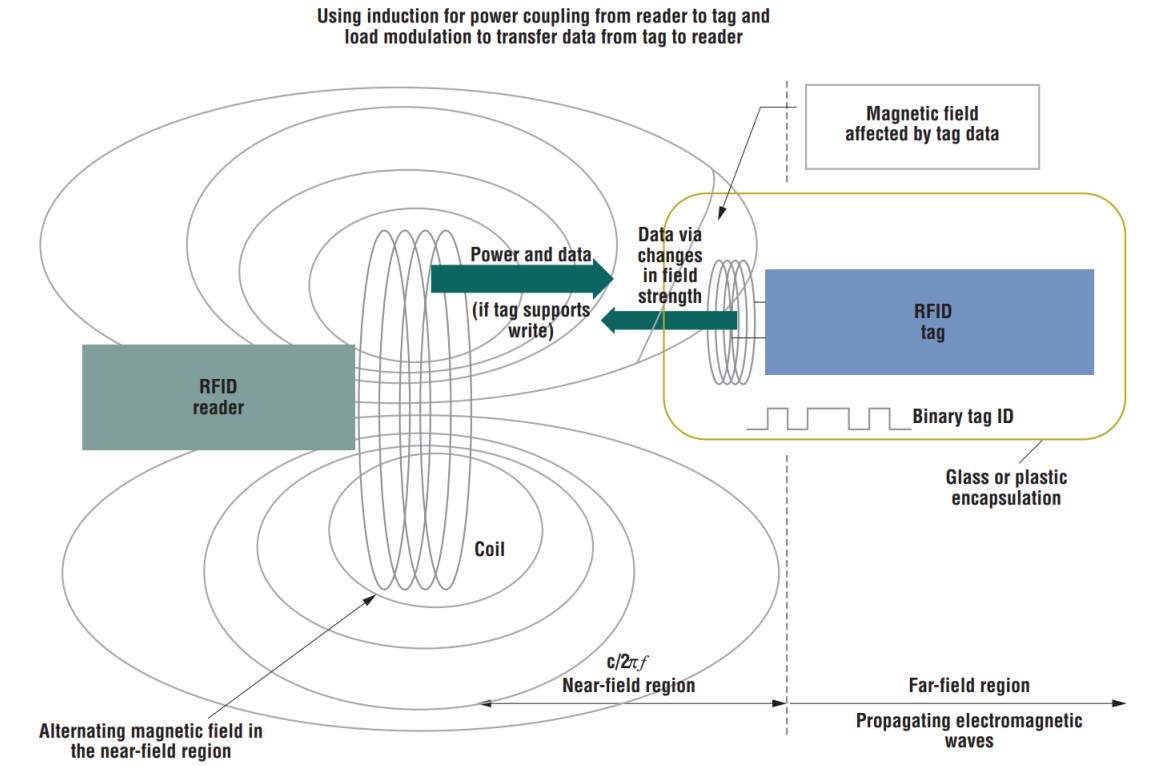

Figure 2.1: Operation of near-field RFID [1]

The subject of this report, the DESFire EV1, is a High-Frequency card.

2.1.2 ISO/IEC 14443

ISO/IEC 14443 is a set of international standards for High-Frequency cards. It is split into 4 parts:

- ISO/IEC 14443-1 : Physical characteristics
- ISO/IEC 14443-2 : Radio frequency power and signal interface
- ISO/IEC 14443-3 : Initialization and anticollision
- ISO/IEC 14443-4 : Transmission protocol

Throughout these standards documents, the term "Proximity Coupling Device" (PCD) is used in place of "reader", and the term "Proximity Integrated Circuit Card" (PICC) is used in place of "card".

2.1.3 Use of RFID in public transport

RFID is widely used in public transportation ticketing as a means of preventing fraud [3]. RFID chips are embedded in existing paper or plastic tickets to act as the primary or secondary proof of a ticket's validity.

Systems with barriers

Figure 2.2: Automatic ticket barriers in the Brussels Metro system [4]

In transport systems with barriers, users must scan their RFID ticket in order to be allowed entrance to the platform or station. The barrier will only open if the ticket is valid (enough credit or journeys remaining etc.). An example of a barrier based system is the Brussels Metro (shown in Figure 2.2).

This reduces the necessity for human ticket inspection, as the barriers act as automatic ticket inspectors.

Systems without barriers

In transport systems without barriers, passengers must validate their RFID ticket by "tagging on" at standalone validators before boarding a tram or bus. There are no physical barriers to stop people from boarding without a ticket. As a result, more ticket inspections are required to detect fare evasion than in transport systems with barriers.

Figure 2.3: RFID ticket validator for a barrierless system in the Netherlands [5]

Ticket inspectors carry hand-held devices which can scan RFID tickets to determine whether a passenger has a valid ticket or not.

2.2 MIFARE smartcard range

The MIFARE brand of smartcards was originally released in 1994 with the MIFARE Classic [6]. The brand quickly became popular in transportation ticketing, with NXP claiming in 2019 that it had a "77% market share" in the sector, and that MIFARE had been deployed in "over 750 cities worldwide" [7].

The MIFARE range of smartcards are secured memory tags - this means that keys are required to perform certain actions on the tag such as reading/writing data, preventing manipulation from unauthorised parties.

2.3 Previous research

2.3.1 MIFARE Classic

The MIFARE Classic has been the subject of much security research. The card used a secret encryption algorithm with 48-bit keys called CRYPTO1. The details of this algorithm were reverse-engineered by German researchers by taking microscopic photos of the circuitry on the card [8]. This determined that the Pseudorandom Number Generator (PRNG) used in the card was based on a 16-bit Linear-Feedback Shift Register (LFSR) which reduced the brute-force space dramatically. Separately, researchers from Radboud University also found attacks on the MIFARE Classic, which was due to be rolled out in the Netherlands as a transportation card (the OV-Chipkaart) [9].

Faster attacks, which require only access to the card [10] have resulted in NXP withdrawing the MIFARE Classic from sale.

2.3.2 MIFARE DESFire

The MIFARE DESFire was released in 2002, bringing support for DES and 3DES encryption. In 2011 however, a paper was published describing a side-channel attack on the DESFire [11]. By performing power analysis using equipment costing approximately $3000, researchers were able to fully recover the master key to the card. While this attack takes a significant amount of time, requiring 250,000 traces, NXP recommends that customers upgrade to the MIFARE DESFire EV1 [12].

2.3.3 MIFARE DESFire EV1

The MIFARE DESFire EV1 was released in 2006. It differs from the predecessor DESFire card by its use of a True Random Number Generator (TRNG) and support for AES-128 encryption modes. It is also certified to Common Criteria EAL4+ level.

To our knowledge, there have been no published attacks against the DESFire EV1 in the literature to date. However, some research has been published investigating possible bias in the TRNG [13]. While the researchers found that the DESFire EV1 TRNG was indeed biased, no practical attacks that exploit this bias have yet been published, to our knowledge.

3 Research Methodology

This chapter discusses the project's aims, and the methodology followed in researching the card, detailing the hardware and software utilised.

3.1 Project Aims

- Investigate and understand the operation of the DESFire EV1
- Create simulated transport applications for the DESFire EV1
- Investigate possible attacks on these simulated applications
- Develop a relay device to carry out Proof of Concept of attacks

3.2 Access to DESFire EV1 documentation

NXP Semiconductors make available a limited set of public datasheets with information about the DESFire EV1. This information consists of general descriptions of the card's functionality, but does not detail the exact commands required to interact with the card in any way.

In order to obtain a full copy of the DESFire EV1 documentation, a customer must sign a non-disclosure agreement (NDA) with NXP. This agreement would restrict the customer from publishing details about the functionality of the card.

For the purposes of this report, no agreement has been signed with NXP. All information detailed herein has been obtained from publicly available research papers, data sheets and the libfreefare library [14].

3.3 Hardware

3.3.1 Test DESFire EV1 cards

Blank MIFARE EV1 cards (model number MF3ICD81) were purchased from an authorised reseller [15] [16]. These cards ship with a default master key, allowing the creation of new applications. This enables the creation of simulated transport apps for the project, the behaviour of which are observed in order to find attacks in the DESFire EV1's application protocol.

3.3.2 Proxmark3

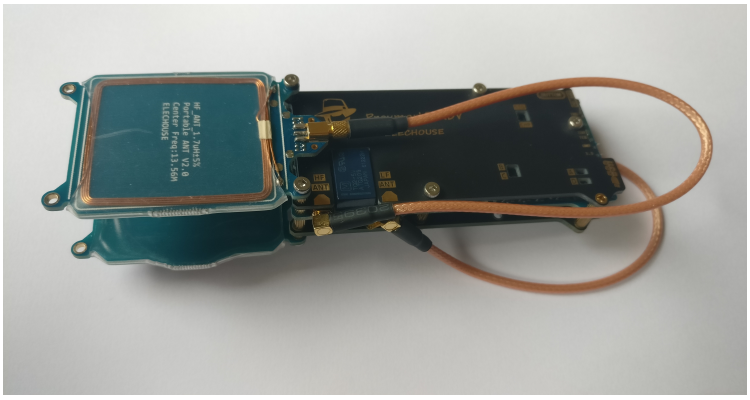

Figure 3.1: Proxmark3

The Proxmark3 is an RFID tool used by penetration testers and researchers. Originally created by researcher Jonathan Westhues [17], it is a completely open-source design, with both the schematics and source code for the firmware available on Github. As depicted in Figure 3.1, it consists of an ARM micro-controller accompanied by an FPGA. Two antennas are included to allow communication with either High Frequency (13.56 Mhz) or Low Frequency (125 kHz) cards. The FPGA modulates and demodulates the signals necessary to communicate with a wide variety of smartcards. It is available for purchase for approximately $200 USD from a variety of manufacturers.

Developers have programmed various routines into the firmware of the Proxmark3. For example, one routine performs simple cloning of basic ID badges, while others carry out the MIFARE Classic attacks described in Section 2.3.1 in order to retrieve the master keys to a card.

Importantly, developers have implemented the necessary functions for encoding and decoding communications with ISO-14443 Type A and Type B tags. As the DESFire

EV1 card is a Type A tag, the `hf 14a snoop` command can be used to snoop on communications between a reader and the card. The High Frequency antenna is placed between the reader and card, capturing packets as they transit. Once complete, a dump of the captured packets can be retrieved from the Proxmark3's memory using the `hf list 14a` command.

3.3.3 SCL3711 NFC Reader

Figure 3.2: SCL3711 NFC Reader

The SCL3711 reader is a USB NFC Reader manufactured by Identive. It utilises a NXP PN533 reader chipset. This chipset is fully compatible with libnfc (see Section 3.4.1). While the PN533 chip can be used in "target mode" to emulate a DESFire EV1 card, it is restricted as it can only emulate cards prefixed with `0x08` in the first byte of the UID of the card.

3.4 Software

3.4.1 libnfc

Libnfc is an open-source C library providing a low-level abstraction layer for interacting with NFC readers [18]. It is compatible with the PN533 chipset used by the SCL3711, and is a dependency for libfreefare. It is used in this project as a "mole" in the Relay Device (see Chapter 5).

3.4.2 libfreefare

This project makes extensive use of the libfreefare library[14]. Libfreefare is an open-source library which provides C APIs for interacting with a wide variety of MIFARE cards including the DESFire EV1. The authentication and cryptographic procedures required for interacting with the DESFire EV1 have been implemented in user space in this library, making it a valuable source of information on the inner workings of the card.

Libfreefare is also used in this project to create simulated transport applications for the DESFire EV1, using the SCL3711 as a reader.

4 MIFARE DESFire EV1

This chapter presents an overview of the functionality of the MIFARE DESFire EV1, based on information from the libfreefare source code[14], public NXP datasheets [19] and public research papers, cited where relevant.

4.1 File Structure

Fundamentally, the DESFire EV1 is an cryptographically authenticated file system. A hierarchy applies as follows: a card can have many applications; applications can have many files and keys. A file has four permission levels (Read Only, Write Only, Read & Write, Change Permissions) which are each assigned to one key). An example layout is shown in Figure 4.1.

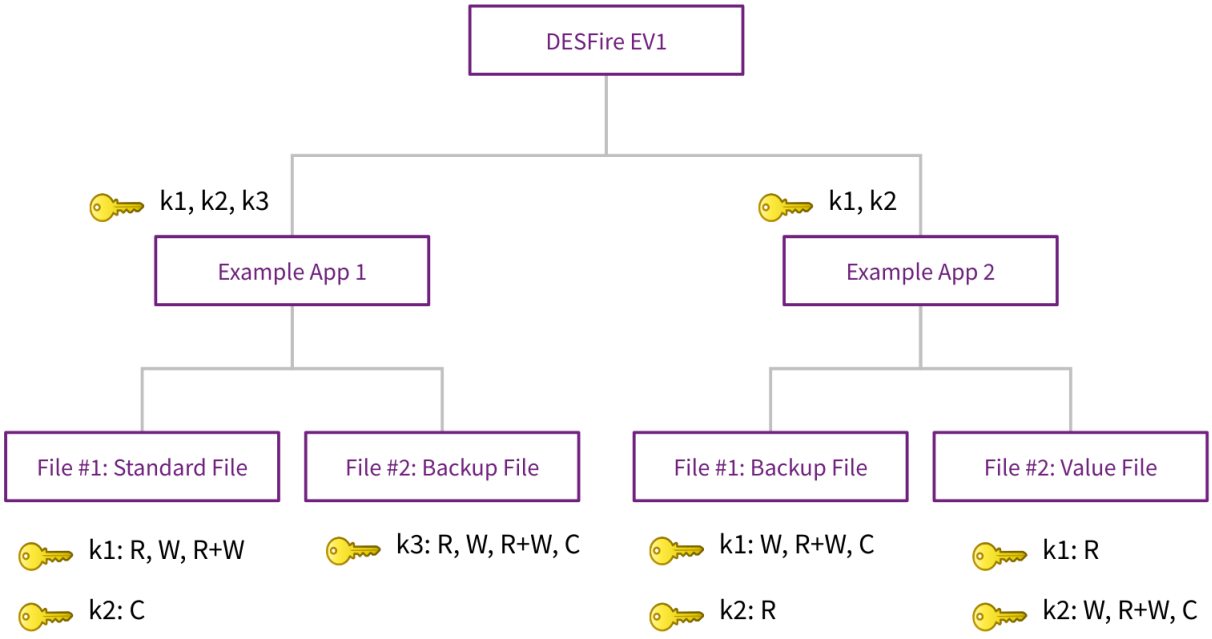

Figure 4.1: Sample file structure of MIFARE DESFire EV1

There are several filetypes that can be used - some of which have an anti-tearing mechanism. "Tearing" describes the phenomenon of a passive RFID card suddenly losing power, due to the removal of the card from the magnetic field. When the anti-tearing mechanism is used, data is written and read from a temporary "mirror" of the permanently stored data. A "Commit" command must be issued before changes to the file are written through from this "mirror" area to non-volatile memory.

Standard Files
This type of file allows raw bytes to be written to and read to non-volatile memory. No anti-tearing mechanism is possible with a Standard File.

Value Files
This type of file stores a numeric value, and is intended for use as a stored balance in a top-up system. Value Files must use the anti-tearing mechanism. This type of file is explored in more detail in Section 6.1.

Backup Files
This type of file is the same as Standard Files, except it has an anti-tearing mechanism. A Commit command must be issued before changes to the data are written to non-volatile memory.

Cyclical Record Files
This type of file allows multiple "records" to be stored. When the file has reached the maximum number of records, the oldest record is overwritten. An anti-tearing mechanism is also used with this file type.

4.2 Authentication Procedure

After selecting an application, the reader and card must carry out an authentication procedure. This mutually authenticates both the reader and card as possessing a particular key. It also serves to generate a unique session key, which will be used to either encrypt communications or generate a MAC, depending on the communication mode used (described in further detail in Section 4.3).

This authentication procedure has been reverse engineered by researchers who developed the Chameleon, an open-source design for hardware that can emulate the MIFARE Classic and DESFire [20].

In Legacy authentication, shown in Figure 4.2, both card and reader pick independent random 64-bit nonces, then seek to prove to each other that they can decrypt encrypted versions of each other's nonce. The decrypted nonces are rotated right or left by 8 bits

before being returned the other party for verification. This step ensures that a Man-In-The-Middle would be unable to extract correct decryptions and reuse them in another authentication session.

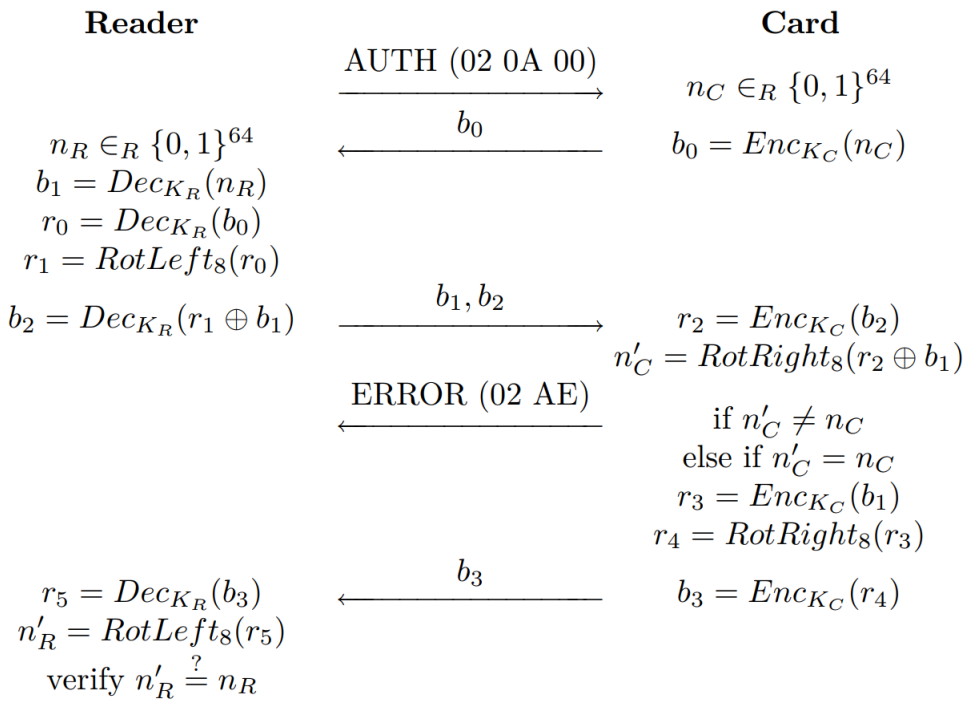

Figure 4.2: Legacy MIFARE DESFire authentication [20]

In Modern authentication, shown in Figure 4.3, a procedure vastly similar to Legacy authentication is performed. In Modern authentication, a nonce size of 128-bits is used, and both card and reader perform encryption and decryption operations, in contrast to Legacy authentication.

In both Legacy and Modern authentication, after the authentication procedure is carried out, a session key is derived from a combination of the reader's and card's random nonces.

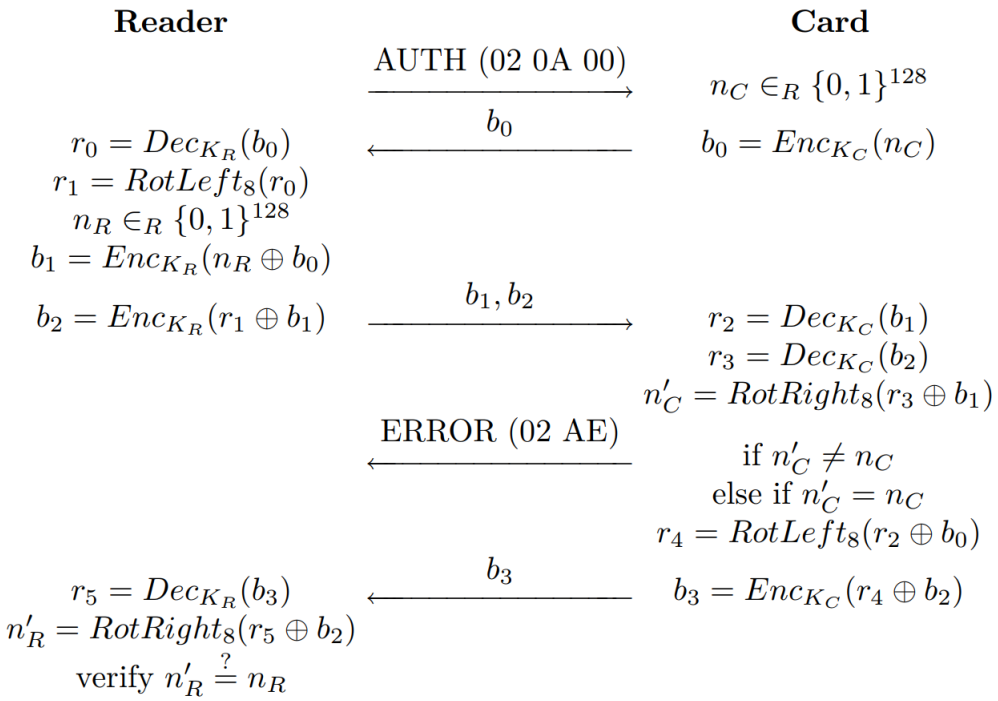

Figure 4.3: Modern MIFARE DESFire EV1 authentication [20]

4.3 Communication Modes

Once authentication has taken place, one of several different communication modes can be used: **Plain**, **Enciphered** or **MAC'd**. The chosen communication mode varies by the key type and authentication mode used (see Table 4.1).

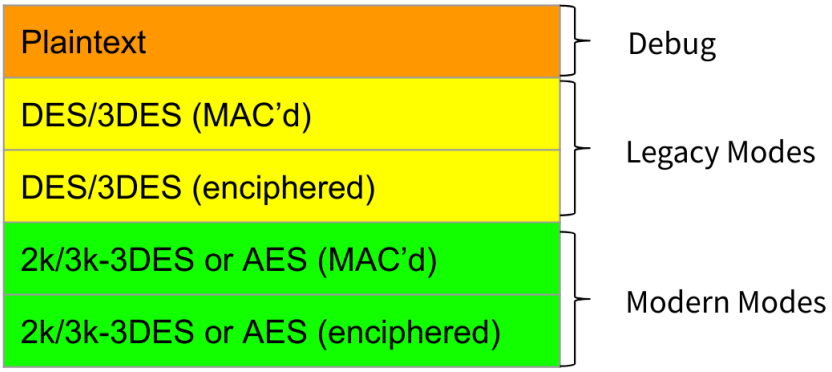

Table 4.1: DESFire EV1 Communication modes

The session key derived from the authentication procedure in Section 4.2 is used to either encipher communications between the reader and card or to generate a Message Authentication Code (MAC) which authenticates the plaintext as genuine.

The block cipher mode used for enciphering is Cipher Block Chaining (CBC mode), and the MAC type is a CBC-MAC.

5 Relay Development

This chapter outlines the development of an RFID relay device in order to inspect and modify ISO 14443 packets as they transit between a genuine reader and card. This allows us to carry out attacks at a protocol level on the DESFire EV1.

5.1 Intro to RFID Relay Attacks

Relay attacks typically involve a "Mole" and "Proxy" device that relays an RFID communication over some distance using wired or wireless means. Typically relay attacks are used to extend the range of an RFID card to "digitally pickpocket" a victim's credit card or access badge [21].

The typical structure of a relay device is shown in Figure 5.1. A "Mole" is used to query the victim's card over RFID. The resulting communication is relayed over wired or wireless means to a "Proxy", which emulates the victim's card to the genuine reader.

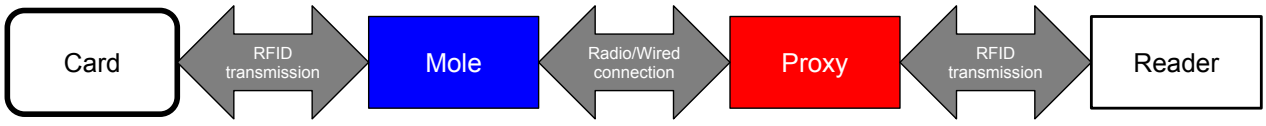

Figure 5.1: Traditional RFID Relay

For this project, a relay device was needed that could alter and replay packets as they passed through the relay device. The relay device developed is shown in Figure 5.2.

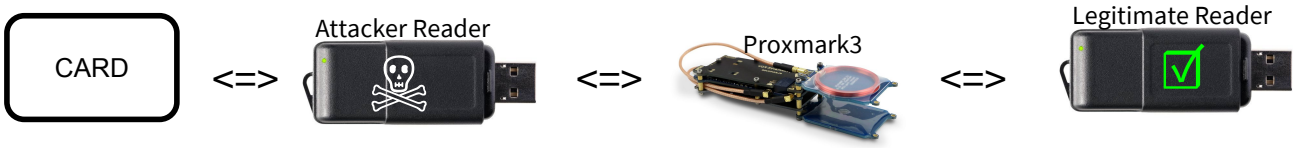

Figure 5.2: Relay device developed for this project

5.2 Python Coordinator

A Python program was written to coordinate the various components of the relay device, and to alter and replay packets as they passed through the relay device. The program communicates with the libnfc mole and the Proxmark3's command line program over USB.

5.3 Libnfc Mole

Libnfc was used to allow the SCL3711 to function as the mole. A simple program was created to send raw reader commands from the Python Coordinator to the card via the SCL3711.

5.4 Modifications to Proxmark3 firmware

Two additional procedures were created during this project in the firmware of the Proxmark3 to enable it to function as a relay device.

5.4.1 `RelayIso14443aTag`

This procedure responds to the ISO 14443 Type A anti-collision and card selection process, and waits for the reader to issue a command. The raw command bytes is returned from the procedure for processing by the Python Coordinator.

5.4.2 `RespondIso14443aTag`

This procedure responds to the reader with the card's raw response which has been relayed to it by the Python Coordinator. The procedure then waits for further

commands from the reader, which it relays back to the Python Coordinator.

5.5 Timing constraint difficulties

Several timing difficulties were encountered during the development of the relay. These were resolved by increasing the FWI (Frame Waiting-time Integer) in the ATS (Answer to Select) returned by the Proxmark3 during the anti-collision and card-selection process. This tells the reader to increase the timeout for card responses, allowing the Proxmark3 greater processing time.

6 Attack 1 - Credit Command Replay

6.1 Value Files and the Credit Command

A Value File is a file format that stores a numeric value on a card. This value can interacted with using several commands:

- GetValue
- Credit
- LimitedCredit
- Debit

The "Credit" and "Debit" commands above have a single argument - the amount to credit or debit the numeric value by. It should be noted that the "Credit" and "Debit" commands are non-idempotent - e.g. a reader sending "Credit 3" once to the card would increment the the balance by 3, whereas a reader sending "Credit 3" twice would increment the balance by 6.

6.2 Credit Application

In order to explore the protocol and features of the DESFire EV1 in the context of public transport, a basic credit application was created for the card using libfreefare[14]. This application credits a ValueFile by 2, mimicking a customer topping up their card at a vending machine. An abbreviated extract from the source code of the application is shown in Listing 6.1.

Listing 6.1: Credit Application

```
1  #define FILENO 1
2  #define CREDIT_AMOUNT 2
3
4  mifare_desfire_select_application (tag, app_id);
5  mifare_desfire_authenticate (tag, FILENO, key);
6  mifare_desfire_credit (tag, FILENO, CREDIT_AMOUNT);
7  mifare_desfire_commit_transaction (tag);
8
9  int32_t balance;
10 mifare_desfire_get_value(tag, FILENO, &balance);
11 printf("Balance is now %d\n", balance);
```

6.3 Observations

Using the Proxmark3's built-in `hf 14a snoop` command to snoop on packets transiting between the genuine SCL3711 reader running the credit application in Listing 6.1 and the card, several traces were collected in a number of different scenarios.

The trace in Figure 6.1 shows Session #1, a trace of the interaction between the reader and card after running the program described in Listing 6.1.

The trace in Figure 6.2 shows Session #2, a trace of the interaction between the reader and card after running the program in Listing 6.1 for a second time.

The difference between the observed Credit Commands in Figure 6.1 and Figure 6.2 is noteworthy. While the command code `0x0C` is unencrypted, the main bodies of the commands are unencrypted. This is because they are two distinct sessions, with two different session keys - thus the command bodies are encrypted to different values (`54 4b fa e6 d1 c6 51 55` and `36 b2 3a cb b1 28 ec af` respectively). This prevents an attacker from using recorded encrypted commands from one session and replaying them in another session.

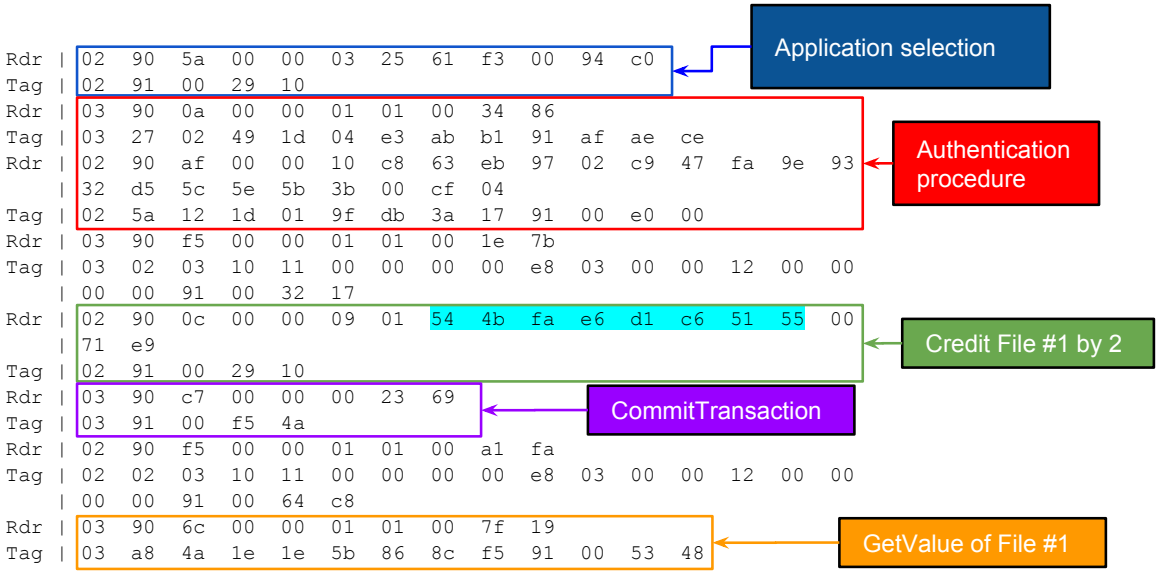

Figure 6.1: Proxmark3 Trace of Credit Application - Session #1

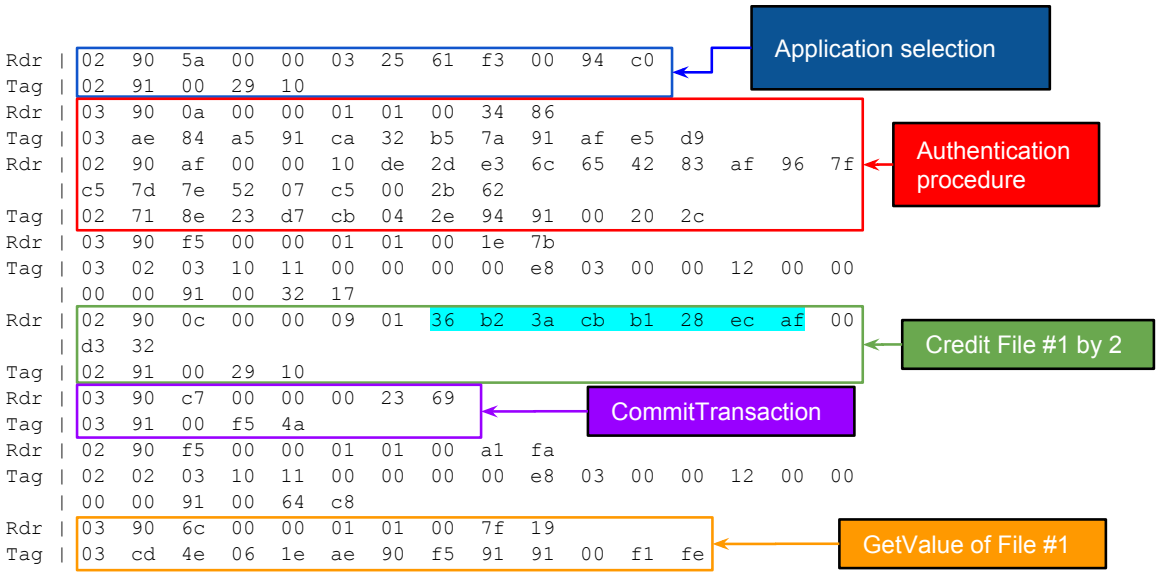

Figure 6.2: Proxmark3 Trace of Credit Application - Session #2

However if we modify the procedure to issue the Credit command *twice* in the same authentication using the code shown in Listing 6.2, it is observed that the encrypted body of both credit commands are equal (`c0 63 1b 18 b6 f4 8d a4`)

Listing 6.2: Modified Credit Application

```
1  #define FILENO 1
2  #define CREDIT_AMOUNT 2
3
4  mifare_desfire_select_application (tag, app_id);
5  mifare_desfire_authenticate (tag, FILENO, key);
6  mifare_desfire_credit (tag, FILENO, CREDIT_AMOUNT);
7  mifare_desfire_credit (tag, FILENO, CREDIT_AMOUNT);
8  mifare_desfire_commit_transaction (tag);
9
10 int32_t balance;
11 mifare_desfire_get_value(tag, FILENO, &balance);
12 printf("Balance is now %d\n", balance);
```

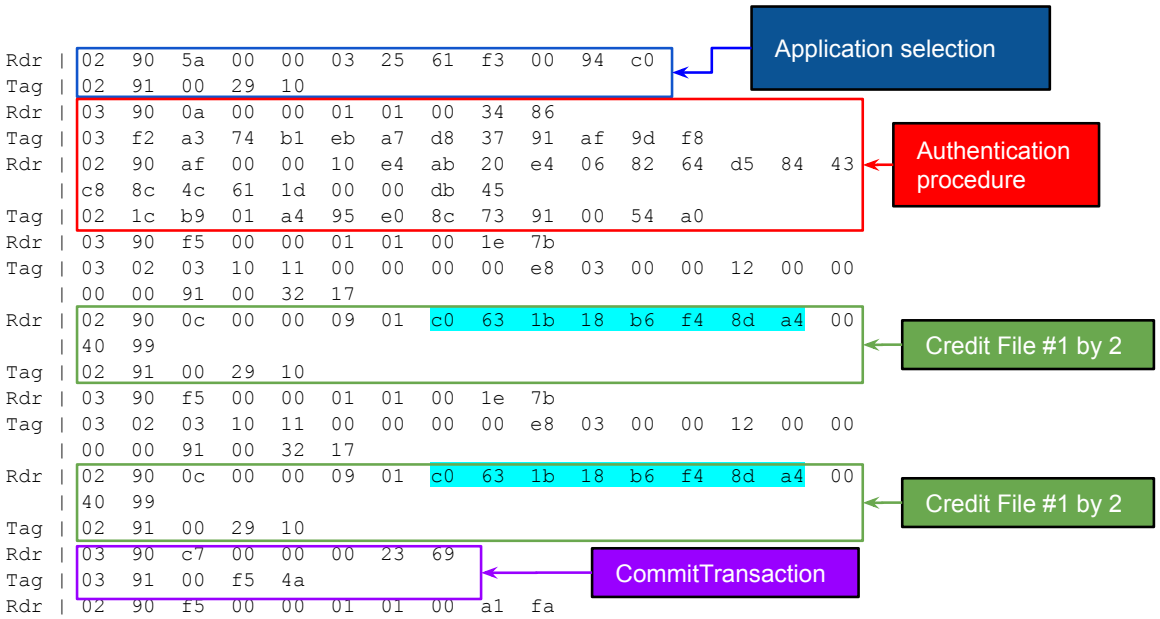

Figure 6.3: Proxmark3 Trace of Modified Credit Application

This means that an attacker can replay Credit commands multiple times in a particular session to increase their effect. As already mentioned, Credit commands are not idempotent, so this will have a cumulative effect on the balance in a ValueFile.

6.4 Example Attack Scenario

1. Approach a top-up machine with Relay device and genuine card
2. Proceed to genuinely top up by a certain amount (e.g €1)
3. When a "Credit" command is received, allow it through to the card, but also store it in memory.
4. Do not allow the card to be powered off, the session terminated, or the transaction to be committed.
5. Replay the stored "Credit" command to the card `n` times
6. Allow the transaction to be committed
7. Observe that the balance on the card has increased in value by `n` times greater than the genuine top up amount.

6.5 Affected Modes

Communication mode	Vulnerable to Attack 1
Plaintext	x
DES/3DES (MAC'd)	x
DES/3DES (enciphered)	x
2k/3k-3DES or AES (MAC'd)	
2k/3k-3DES or AES (enciphered)	

Table 6.1: Communication Modes vulnerable to Attack 1

As shown in Table 6.1, only the Legacy modes of communication (DES/3DES MAC'd or enciphered) are affected by this attack. This is due to the resetting of the IV to null (all 0x00 bytes) at the beginning of each command's transmission in Legacy mode. This is in contrast to Modern communication modes, where the IV is not reset at the beginning of each command, but rather is a continuation of the previous chained encryptions from previous commands.

6.6 Proof of Concept

In order to prove this attack, the relay device described in Chapter 5 was used to attack the Credit Application in Listing 6.1. The Python coordinator was used successfully to

detect the use of the Credit command and replay this 10 times more than intended to the card. Later queries showed the balance in the ValueFile had increased by 10 times the intended amount. This proves the feasibility of Attack 1.

7 Attack 2 - Faked OK Response

7.1 Tag-on Application

The WriteData command is used to write bytes to a Standard File or Backup File.

To observe the behaviour of the WriteData command, a simple tag-on application was created using libfreefare [14], which writes an ASCII "H" or "G" to a Standard File on the card, depending on whether or not the card has already been tagged on or not. An abbreviated listing of this program is shown in Listing 7.1.

Listing 7.1: Tag-on Application

```
1  #define FILENO 1
2  #define BYTES_TO_READ 2
3  #define OFFSET 0
4
5  mifare_desfire_select_application (tag, app_id);
6  mifare_desfire_authenticate (tag, FILENO, key);
7
8  char buffer[BYTES_TO_READ];
9  mifare_desfire_read_data(tags, FILENO, OFFSET, BYTES_TO_READ, buffer);
10
11 if(strcmp(buffer, "H") == 0) {
12     // Tagged on
13     char *s1 = "H";
14     mifare_desfire_write_data (tag, FILENO, OFFSET, strlen(s1)+1, s1);
15 } else {
16     // Not tagged on
17     char *s2 = "G";
18     mifare_desfire_write_data (tag, FILENO, OFFSET, strlen(s2)+1, s2);
19 }
```

7.2 Observations

Using the Proxmark3's built-in `hf 14a snoop` command to snoop on packets transiting between the genuine SCL3711 reader running the tag-on application in Listing 7.1, the trace in Figure 7.1 was produced.

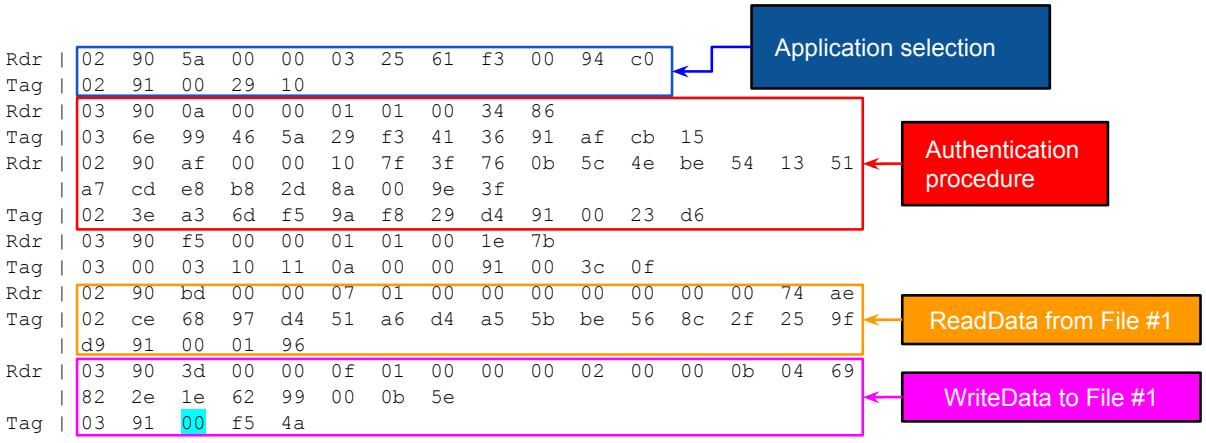

Figure 7.1: Proxmark3 Trace of Tag-on Application

From Figure 7.1, it can be seen that the card's response to the ReadData command is encrypted. However when the reader writes data to the card using the WriteData command, the "OK" response of the card which indicates that the command ran successfully (00) is completely unencrypted.

This means that an attacker could spoof the "OK" response of a WriteData command, without actually running the command on the card. In the context of a transport system with barriers (see Section 2.1.3), an attacker can "tag-on" without actually allowing their balance to be decremented.

When the attacker reaches the end of the journey and tries to leave via an exit barrier, the system will detect that the card has not been tagged on. In some cases, the system might throw an error and require human intervention; in others, the system will charge the maximum possible fare to the user's card. In the latter case, the attacker can perform the same trick by spoofing the "OK" response - which results in no charge actually being made against their card.

7.3 Example Attack Scenario

1. Approach the barrier reader with Relay device.

2. When a WriteData command is received, respond with "OK" to the reader but don't relay the WriteData command to the genuine card. The barriers will open.
3. Keep the card powered on, and the WriteData command cached in memory in case of random ticket inspection during the journey.
4. If not inspected:
   a) Power down the card. The card has not changed balance - it's as if it was never tagged on at the barrier.
   b) Present the Relay device at the exit barrier. As the card appears to not be tagged on, the barrier will try to charge the maximum fare to the card.
   c) When the maximum fare WriteData command is intercepted, simply respond with "OK" to the exit barrier reader, and do not relay the command to the genuine card.
   d) Exit through the opened gates. The attacker has travelled for free, and the genuine card still has retained its original balance.
5. If inspected:
   a) Allow the WriteData command through to the genuine card
   b) Present the genuine, tagged on card for inspection. The attacker has avoided being fined, but their balance has been decremented.

7.4 Affected modes

Communication mode	Vulnerable to Attack 2
Plaintext	x
DES/3DES (MAC'd)	x
DES/3DES (enciphered)	x
2k/3k-3DES or AES (MAC'd)	
2k/3k-3DES or AES (enciphered)	

Table 7.1: Communication Modes vulnerable to Attack 2

As shown in Table 7.1, only the Legacy modes are vulnerable to Attack 2. Under Modern modes, responses from the card are encrypted or MAC'd depending on the communication mode used. This allows the reader to verify that the command it sent to the card was actually executed, preventing Attack 2.

8 Attack 3 - Suspended Commit

8.1 Modified AES Tag-On Application

The previous two attacks have solely affected the Legacy modes of communication. For this attack, the anti-tearing mechanism was examined in detail in the Modern mode of communication. The Tag-on Application from Listing 7.1 was modified to use AES, a Modern communication mode and to use a Backup File with associated anti-tearing mechanism. An abbreviated version of the modified source code is presented in Listing 8.1.

Listing 8.1: Modified AES Tag-on Application

```
#define FILENO 1
#define BYTES_TO_READ 2
#define OFFSET 0

mifare_desfire_select_application (tag, app_id);
mifare_desfire_authenticate (tag, FILENO, key);

char buffer[BYTES_TO_READ];
mifare_desfire_read_data(tags, FILENO, OFFSET, BYTES_TO_READ, buffer);

if(strcmp(buffer, "H") == 0) {
    // Tagged on
    char *s1 = "H";
    mifare_desfire_write_data (tag, FILENO, OFFSET, strlen(s1)+1, s1);
    mifare_desfire_commit_transaction (tag);
} else {
    // Not tagged on
    char *s2 = "G";
    mifare_desfire_write_data (tag, FILENO, OFFSET, strlen(s2)+1, s2);
    mifare_desfire_commit_transaction (tag);
}
```

8.2 Observations

Using the Proxmark3's built-in `hf 14a snoop` command to snoop on packets transiting between the genuine SCL3711 reader running the tag-on application in Listing 8.1, the trace in Figure 8.1 was produced.

From Figure 8.1, it can be observed that the Commit command is usually the final command in the sequence of a transport app. This is because it must be the final action of the reader to issue the Commit command - otherwise the anti-tearing mechanism would be redundant, as power could be lost in between commands.

If an attacker learns the usual number of commands preceding the Commit command, they can "Suspend" the final command in memory in a relay device, and keep the card powered on. This gives the attacker the ability to decide whether or not the transaction gets run on their card or not.

This attack applies primarily to barrierless systems, as it requires the attacker to not allow the card to fully complete the transaction while at the ticket validator. This means the validator will be in an error state.

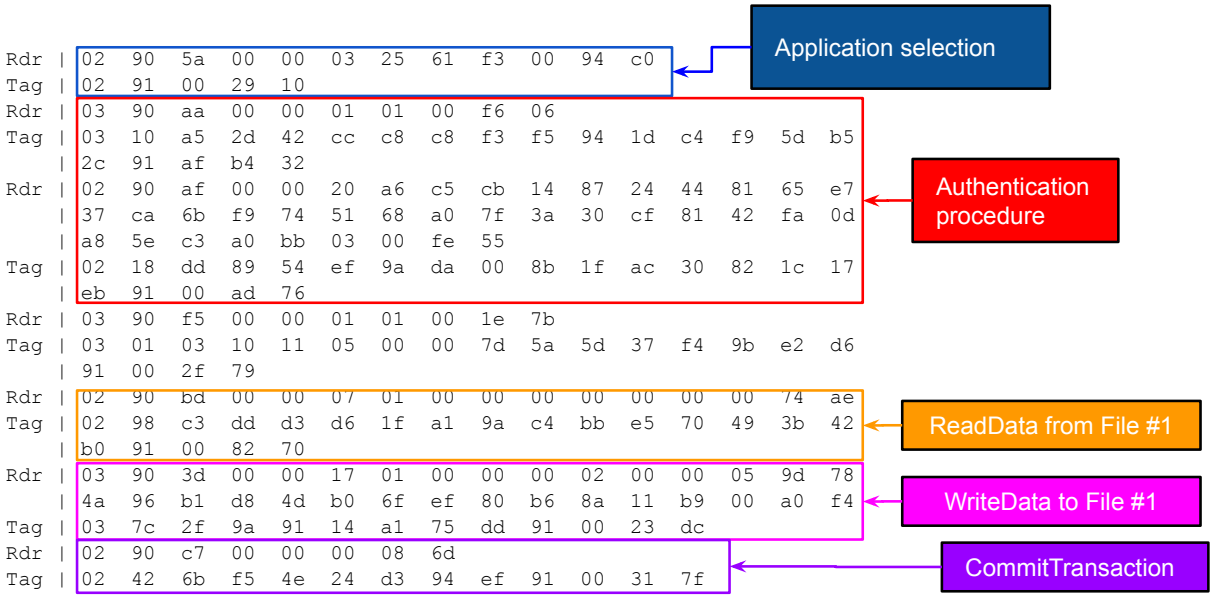

Figure 8.1: Proxmark3 Trace of AES Tag-on Application

8.3 Example Attack Scenario

1. Approach the ticket validator with Relay device.
2. Using prior obtained knowledge about the sequence of commands the validator will issue, allow all commands **except** the final Commit command through to the card.
3. Keep the card powered on, and the Commmit command cached in memory in case of random ticket inspection during the journey.
4. If not inspected:
   a) At the end of the journey, power down the card and discard the cached Commit command. As the anti-tearing mechanism will revert any changes made to Backup Files or Value Files, no change will have been made to the balance on the card.
5. If inspected:

a) Allow the Commit command through to the genuine card. This will write all data to non-volatile memory.

b) Present the genuine, tagged on card for inspection. The attacker has avoided being fined, but has decremented their balance.

8.4 Affected modes

Communication mode	Vulnerable to Attack 3
Plaintext	x
DES/3DES (MAC'd)	x
DES/3DES (enciphered)	x
2k/3k-3DES or AES (MAC'd)	x
2k/3k-3DES or AES (enciphered)	x

Table 8.1: Communication Modes vulnerable to Attack 3

As shown in Table 8.1, all communication modes are affected by this attack. As the attack is more on the logic of the anti-tearing mechanism itself, it has more widespread affect.

9 Evaluation

9.1 Discussion of impact

	Attack 1	Attack 2	Attack 3	
Plaintext	✓	✓	✓	Debug
DES/3DES (MAC'd)	✓	✓	✓	Legacy
DES/3DES (enciphered)	✓	✓	✓	
2k/3k-3DES or AES (MAC'd)			✓	Modern
2k/3k-3DES or AES (enciphered)			✓	

Table 9.1: Summary of communications modes affected by attacks

Table 9.1 shows a summary of the attacks found and the communications modes affected. Attacks 1 and 2 are limited to the Legacy communications modes, while Attack 3 affects all communications modes.

There are some caveats to the feasibility of carrying out Attacks 1 and 2. The use of Legacy communication modes is less widespread as operators upgrade to Modern communication modes. Furthermore, the use of Value Files appears to be almost non-existent among transport operators - it appears the use of Standard and Backup Files to store a balance is much more popular.

Attack 3 affects both Legacy and Modern Authentication modes, and is more of an attack on the anti-tearing mechanism itself. While this attack has the broadest impact, it can be successfully mitigated by backend fraud detection systems.

9.2 Suggested Mitigations

1) Upgrade Legacy mode systems to Modern mode

Transportation systems whose ticketing systems utilise the ValueFile functionality of the DESFire EV1 with a Legacy communication mode should particularly seek to upgrade their systems to a Modern communication mode. This would completely negate the effect of Attack 1, as Modern communication modes do not reset the Initialisation Vector (IV) before each command, preventing an attacker from replaying observed Credit commands.

2 a) Backend fraud detection

Transaction logs should be automatically sent to backend systems, where they can be used to correlate genuine transactions with the observed balance of cards. Where anomalies are detected, cards should be blacklisted by their unique identifier (UID).

2 b) Use key diversification

Blacklisting cards by their UID as above would be futile if the same encryption key is used for all cards, as an attacker could easily spoof a different UID. Key diversification uses different keys for different card UIDs, which allows more effective banning of specific card UIDs.

10 Conclusion

10.1 Reflection on Project Aims

The project aims were:

- Investigate and understand the operation of the DESFire EV1
- Create simulated transport applications for the DESFire EV1
- Investigate possible attacks on these simulated applications
- Develop a relay device to carry out Proof of Concept of attacks

All of the above aims have been carried out in this project, however only one Proof of Concept attack was created, for Attack 1. Proof of Concepts for Attacks 2 and 3 should be possible in future work.

10.2 Disclosure

Details of the attacks as outlined in this report will be sent to NXP's Product Security Incident Response Team (PSIRT).

10.3 Future Work

As mentioned, Proof of Concepts for Attacks 2 and 3 should be possible to create in future work.

There is also ample room for further research on the DESFire EV1. For example, this project did not explore possible cryptographic attacks on the card's communications. The card's usage of a CBC-MAC as authentication for commands is particularly ripe for investigation, as are possible bit-flipping attacks on the enciphered communications modes used.

Bibliography

[1] Roy Want. An Introduction to RFID Technology. *IEEE Pervasive Computing*, (1):25–33, 2006.

[2] Klaus Finkenzeller. *RFID Handbook: Fundamentals and Applications in Contactless Smart Cards, Radio Frequency Identification and Near-Field Communication*. John Wiley & Sons, 2010.

[3] Keith E. Mayes, Konstantinos Markantonakis, and Gerhard Hancke. Transport ticketing security and fraud controls. *Information Security Technical Report*, 14(2):87 – 95, 2009. Smart Card Applications and Security.

[4] Ingolf. Bruxelles / Brussel - Metro - Beekkant. `https://commons.wikimedia.org/wiki/File:Bruxelles_-_Brussel_-_Metro_-_Beekkant_(13190233824).jpg`, Retrieved 21st April 2019.

[5] Ischa1. Een OV-chipkaart poortje op het station van Amersfoort. `https://commons.wikimedia.org/wiki/File:OV-chipkaartpoortjeopstationAmersfoort.jpg`, Retrieved 21st April 2019.

[6] Keith E Mayes and Carlos Cid. The Mifare Classic Story. *Information Security Technical Report*, 15(1):8–12, 2010.

[7] NXP Semiconductors B.V. MIFARE - The Leading Brand of Contactless IC Products. `https://www.mifare.net/en/about-mifare/`, Retrieved 18th April 2019.

[8] Karsten Nohl and Henryk Plötz. Mifare, little security, despite obscurity. In *the 24th Congress of the Chaos Computer Club in Berlin, December 2007*, 2007.

[9] Gerhard de Koning Gans, Jaap-Henk Hoepman, and Flavio D Garcia. A practical attack on the mifare classic. In *International Conference on Smart Card Research and Advanced Applications*, pages 267–282. Springer, 2008.

[10] Nicolas T. Courtois. The Dark Side of Security by Obscurity and Cloning MiFare Classic Rail and Building Passes Anywhere, Anytime. Cryptology ePrint Archive, Report 2009/137, 2009. https://eprint.iacr.org/2009/137.

[11] David Oswald and Christof Paar. Breaking Mifare DESFire MF3ICD40: Power Analysis and Templates in the Real World. In Bart Preneel and Tsuyoshi Takagi, editors, *Cryptographic Hardware and Embedded Systems – CHES 2011*, pages 207–222, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[12] NXP Semiconductors B.V. Update on MIFARE DESFire (MF3ICD40). https://www.mifare.net/update-on-mifare-desfire-mf3icd40/, Retrieved 17th April 2019.

[13] D. Hurley-Smith and J. Hernandez-Castro. Certifiably Biased: An In-Depth Analysis of a Common Criteria EAL4+ Certified TRNG. *IEEE Transactions on Information Forensics and Security*, 13(4):1031–1041, April 2018.

[14] libfreefare — github. https://github.com/nfc-tools/libfreefare, Retrieved 15th April 2019.

[15] Dot Origin Ltd. Smartcard Focus. https://www.smartcardfocus.com, Retrieved 16th April 2019.

[16] NXP Semiconductors B.V. MIFARE Company List - Dot Origin Ltd. https://www.mifare.net/company/dot-origin-ltd/, Retrieved 18th April 2019.

[17] Jonathan Westhues. A Test Instrument for HF/LF RFID. 2007. http://cq.cx/proxmark3.pl, Retrieved 14th April 2019.

[18] libnfc — github. https://github.com/nfc-tools/libnfc, Retrieved 15th April 2019.

[19] NXP Semiconductors. MF3ICDx21 41 81-MIFARE DESFire EV1 contactless multi-application IC. *Rev*, 3.2:145632, 2015.

[20] Timo Kasper, Ingo Von Maurich, David Oswald, and Christof Paar. Chameleon: A versatile emulator for contactless smartcards. In *International Conference on Information Security and Cryptology*, pages 189–206. Springer, 2010.

[21] Gerhard P Hancke. A practical relay attack on ISO 14443 proximity cards. *Technical report, University of Cambridge Computer Laboratory*, 59:382–385, 2005.

A1 Appendix

A1.1 Associated Code

Code associated with this project is located on the CD attached to the rear cover of this booklet. This includes:

- Modified Proxmark3 Firmware Source Code
- Proof of Concept Code for Attack 1
- Simulated Transport Applications for Attacks 1,2,3