A Comprehensive Study on the RFID Technology of Delhi Metro Cards

Vanita Jain[1], Rishab Bansal[2] and Mahima Swmai[3] and Achin Jain[4]

[1]Bharati Vidyapeeth's College of Engineering, INDIA; vanita.jain@bharatividyapeeth.edu
[2]Bharati Vidyapeeth's College of Engineering, INDIA; rishabbansal.it1@bvp.edu.in
[3]Bharati Vidyapeeth's College of Engineering, INDIA; mahima.it1@bvp.edu.in
[4]Bharati Vidyapeeth's College of Engineering, INDIA; achin.mails@gmail.com

* Correspondence: vanita.jain@bharatividyapeeth.edu

Abstract

In this paper, the authors analyse RFID technology, different types of Tags, Readers and various protocols associated with RFID. We read, write and dump the raw bytes from the MIFARE Classic Card using RC5220 and Arduino Mega. Delhi Metro Card uses MIFARE DESFire. MIFARE DESFire uses 3(DES) for encryption. In this paper, we also read the data structure of the Delhi Metro Card. Additionally, we also compare MIFARE Classic Card and MIFARE DESFire Card.

Keywords: RFID, Smart Cards, Arduino, MIFARE Cards

1.Introduction

Radio-Frequency Identification (RFID) is a technology that can be used to identify & track tags/cards automatically. An RFID tag has three parts, a radio receiver, a transmitter & a radio transponder. When the card is moved near to the reader, the coil around the transmitter is charged by the electromagnetic pulse from the reader and makes it transmit the digital data to the reader. This data can be an ID or a Serial Number which can be used to keep track of goods, people, inventory, collecting toll etc[1][2]. There are 2 types of RFID Tags: Active Tags are powered by a battery and therefore can be used to send data at a longer range (100s of meters) from the RFID reader. Passive Tags do not have any battery attached, they get power from the electromagnetic pulses from the RFID reader which limits their use cases to situations where the tag has to be placed very close to the Reader. Fig 1 shows a typical RFID system.

DOI: 10.5281/zenodo.5171426

Received December 30, 2020 Accepted Jun 01, 2021

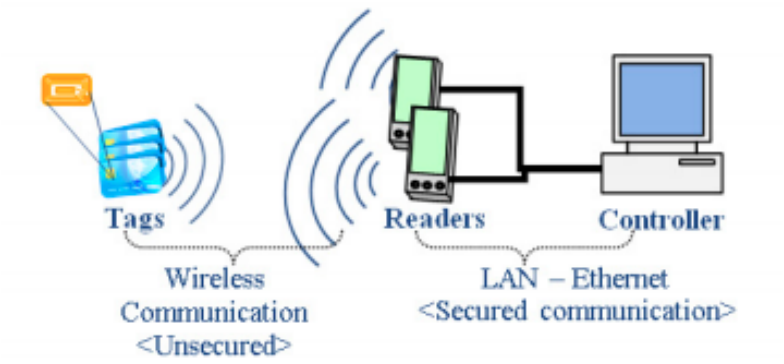

Fig 1. Typical RFID System

The goal of Automatic Identification & Data Capture can also be achieved by Barcodes[3] or QR codes[4]. Barcodes & QR Codes are also used in a lot of similar use cases, but they require line of sight to work. RFID tags don't require a direct line of sight. They just need to be in the proximity of the reader. Close for Passive Tags, 100s of meters for Active readers.
Delhi Metro Cards use RFID technology for Entering, Exiting, Recharging etc. Our paper is a compilation of the analysis on RFID Technology in general and its implementation in MIFARE Classic Cards & MIFARE DESFire Cards (Delhi Metro Cards).

2. RFID Design

2.1 Tags

RFID Tags consists of three components: an antenna (used for receiving or transmitting the signal), a substrate & a microchip (an Integrated Circuit (IC) which is used to store the information, process it, generate the Radio-Frequency (RF) signals)[5][6]. The RFID Tag can either have a programmable or fixed Logic to process the data and transmit the data.
Tags can also be classified based on their read-write properties. A read-only Tag will have only a factory-assigned Unique-Identification(UID) or a Serial Number which can be used as a key for Doors, Databases. Read-Write tags can be re-written by the system user for future upgrades or updates. Write-once-Read-multiple Tags can be used in Markets, Offices etc.

2.2 Readers

Active Reader Passive Tag (ARPT)[7] is an RFID system where the reader is active. The reader transmits interrogator signals. The reader also receives the replies from the Passive Tags.
Active Reader Active Tags (ARAT)[7] is an RFID system that has both Reader and the Tag active. The active Tag in this system is pinged and awoken by the interrogator signal from the Active Reader.
Passive Reader Active Tag (PRAT)[7] is an RFID system where the reader is passive. It only receives radio signals. Active Tag means a battery-operated Tag. A PRAT RFID system can have a range from 1-200 feet.

2.3 Signalling

Signalling among the Tag and Reader can be done in different ways, which depends upon the frequency band being used by the Tag. Tags that operate on Low-Frequency & High-Frequency (Radio Wavelength) need to be very near to the reader antenna. Tags that use Ultra-High-Frequency use a different approach (each scattering) because they are more than 1 radio wavelength away from the reader.[8]

2.4 Frequencies

Table 1[9] shows different RFID Frequency Bands.

DOI: 10.5281/zenodo.5171426

Received December 30, 2020 Accepted Jun 01, 2021

Table I RFID FREQUENCY BANDS

BAND	REGULATIONS	RANGE	DATA SPEED
LF: 120-150 Khz	Unregulated	10 cm	Low
HF: 13.56 MHz	ISM band Worldwide	10cm-1m	Low to Moderate
UHF: 433 MHz	Short-range devices	1-100m	Moderate
UHF: 865-868 MHz (Europe) 902-928 MHz (North America)	ISM band	1-12m	Moderate to High
Microwave: 2450-5800 MHz	ISM band	1-2m	High
Microwave: 3.1-10 GHz	Ultra-wideband	Up to 200m	High

2.5. Internal Architecture

2.5.1. Manufacturer Block

This is the first data block. It contains the Integrated Circuit (IC) manufacturer data. This block is programmed and write protected in the production test. Fig 2 shows the Manufacturer Block.

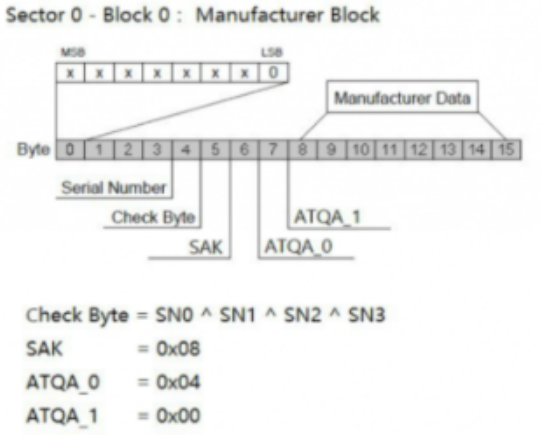

Fig 2. Manufacturer block explained

DOI: 10.5281/zenodo.5171426

Received December 30, 2020 Accepted Jun 01, 2021

Fig 3[10] shows the manufacturer block for the 4-Byte NUID and 7-Byte UID versions respectively.

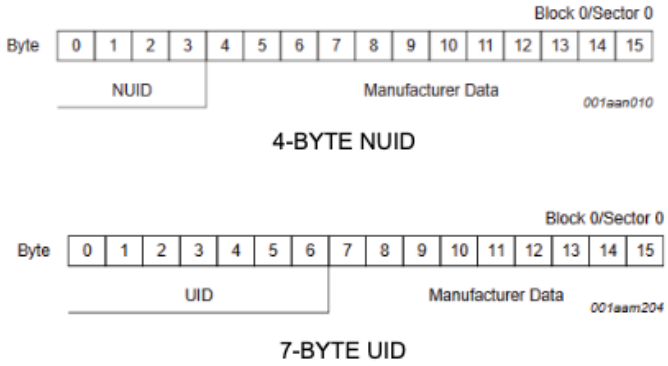

Fig 3. 4-Byte NUID and 7-Byte UID

2.5.2 UID/Serial Number

The first 9 bytes of the memory store the 7-Byte Unique Identification (UID) and 2 Check-Bytes. It covers the pages with addresses 00h, 01h, and 2 bytes of 02h. The Integrated Circuit (IC) manufacturer programs these Bytes. Because of security reasons, these Bytes are Write-Protected. Fig 4[11] shows the UID/Serial Number Data Block.

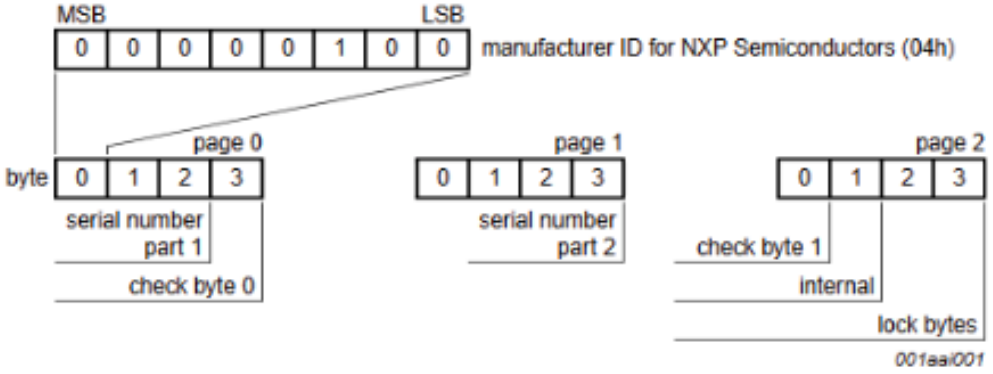

Fig 4. UID/Serial number

2.5.3 Memory Organisation

The 52 bit EEPROM memory is spread across 16 pages having 4 bytes per page. EEPROM cells are read as logic 0(Erased state) and logic 1(written state). Table II[12] shows the Memory Organisation of the EEPROM.

Table II MEMORY ORGANISATION

PAGE ADDRESS		**BYTE NUMBER**			
DECIMAL	**HEX**	**0**	**1**	**2**	**3**
0	00h	Serial Number			
1	01h	Serial Number			
2	02h	Serial Number	Internal	Lock Bytes	Lock Bytes
3	03h	OTP	OTP	OTP	OTP
4 to 15	04h to 0Fh	User Memory			

DOI: 10.5281/zenodo.5171426

Received December 30, 2020 Accepted Jun 01, 2021

3. Experimental Setup

3.1. Module RC522

The RC522 is a 13.56MHz RFID module that is based on the MFRC522 controller by NXP semiconductors. Fig. 5 shows (a)RC522, (b)MIFARE Classic Card, (c)Key Fob(Token). RC522 can also support various communications protocols like I2C[13], SPI[14] and UART[15].

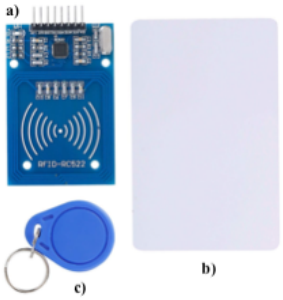

Fig 5. (a)RC522 module, (b)MIFARE classic card, (c)key fob

3.2. Circuit Diagram

Fig 6 shows the circuit diagram of the connections between RC522 and Arduino Mega that we used in our experiments.

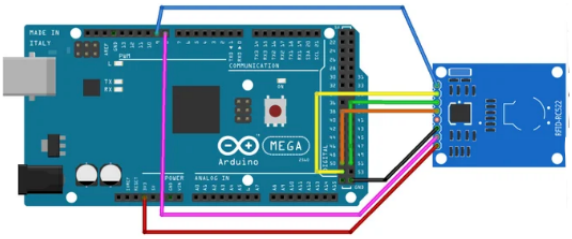

Fig 6. Circuit diagram

3.3. Our Setup

Fig 7 shows our experimental setup and connection.

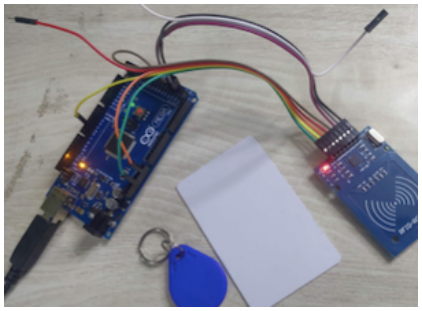

Fig 7. Our setup

3.4. NUID of Card and Token

DOI: 10.5281/zenodo.5171426

Received December 30, 2020 Accepted Jun 01, 2021

Fig. 8 shows the 4-Byte Non-Unique Identifier (NUID) scanned from the MIFARE Classic Card using our script[16].

```
-> A new card has been detected.
-> The NUID tag is:
-> In hex:  07 C9 79 62
-> In dec:  07 201 121 98
```

Fig 8. UID of card

```
-> A new card has been detected.
-> The NUID tag is:
-> In hex:  69 26 7D A2
-> In dec:  105 38 125 162
```

Fig 9. UID of token

Fig. 9 shows the 4-Byte Non-Unique Identifier (NUID) scanned from the MIFARE Classic Token using our script[16].

3.3. Data-Dump of the Card

Fig. 10 shows the Data-Dump of the MIFARE Classic Card which we extracted using our script[17]. Fig 10 also shows that the MIFARE Classic Card has 16 sectors of 64 Bytes each.

```
-> Card UID: 07 C9 79 62
-> Card SAK: 08
-> PICC type: MIFARE 1KB
-> Sector Block   0  1  2  3   4  5  6  7   8  9 10 11  12 13 14 15  AccessBits
->   15     63   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
->          62   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
->          61   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
->          60   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
->   14     59   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
->          58   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
->          57   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
->          56   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
                                    .
                                    .
                                    .
->    0      3   00 00 00 00  00 00 FF 07  80 69 FF FF  FF FF FF FF  [ 0 0 1 ]
->           2   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
->           1   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  [ 0 0 0 ]
->           0   07 C9 79 62  D5 08 04 00  62 63 64 65  66 67 68 69  [ 0 0 0 ]
```

Fig 10. Data dump of MIFARE classic card

3.6. Data-Dump of the Card and the Token

Fig 11 shows the Data Extracted from the MIFARE Classic Card using the Mobile App TagInfo by NXP[18]. Fig 11 also verifies the serial number to the NUID of the MIFARE Classic Card that we found in our experimental setup.

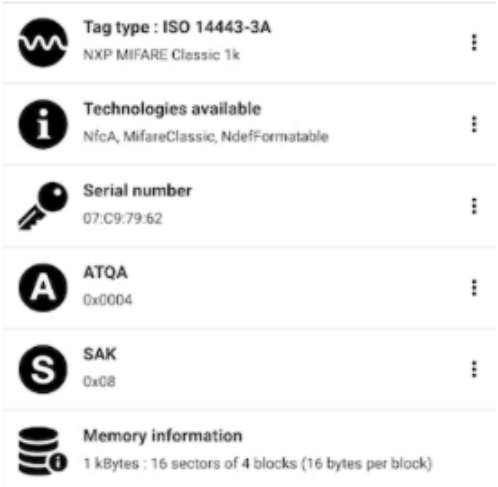

Fig 11. Data extracted using mobile NFC app (card)

Fig 12. Data extracted using mobile NFC app (token)

Fig 12 shows the Data Extracted from the MIFARE Classic Token using the Mobile App TagInfo by NXP[18]. Fig 12 also verifies the serial number to the NUID of the MIFARE Classic Token that we found in our experimental setup.

DOI: 10.5281/zenodo.5171426

Received December 30, 2020 Accepted Jun 01, 2021

4. Data Structure of Delhi Metro Card

The following figures show the detailed Data Structure of the Delhi Metro Card obtained by mobile app TagInfo by NXP[18].

Fig 13 shows the IC Manufacturer and 7 DESFire Applications in the Delhi Metro Card.

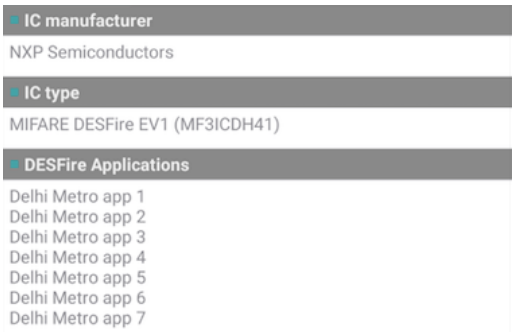

Fig 13. IC manufacturer and DESFire applications

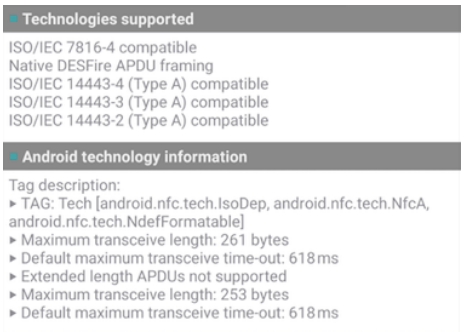

Fig 14. Technologies supported

Fig 14 shows the Technologies supported and Android Technology Information of the Delhi Metro Card.

Fig 15 shows the Memory Information and Version Information of the Delhi Metro Card.

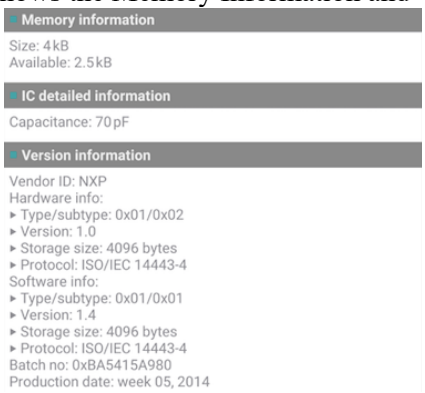

Fig 15. Memory information and version information

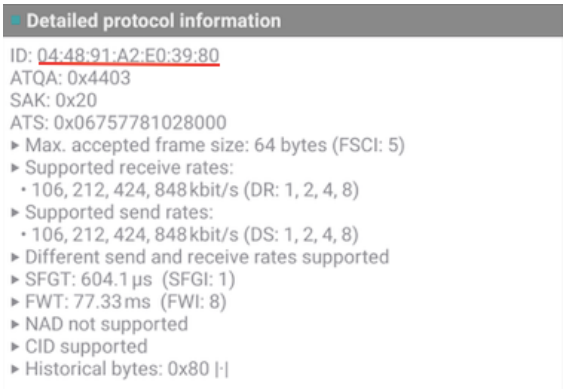

Fig 16. Protocol information

Fig 16 shows the Detailed Protocol Information of the DelhiMetro Card. We can also see that the Delhi Metro Card contains a 7-Bytes(Highlighted) Unique Identifier (UID), whereas the MIFARE Classic Card & Token have only a 4-Byte Non-Unique Identifier (NUID). Fig 16 also shows the supported receive rates and accepted frame sizes.

The following figures show the Memory Content of the Delhi Metro Card.

Fig 17 shows the Application ID 0x000000 (PICC). It also shows that this app contains 1 (3)DES Key and different permissions associated with it.

DOI: 10.5281/zenodo.5171426

Received December 30, 2020 Accepted Jun 01, 2021

Fig 17. Application ID 0x000000 (PICC)

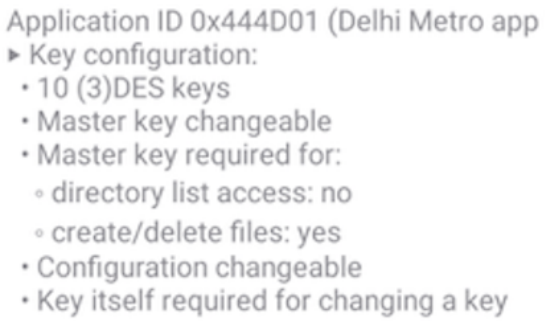

Fig 18. Application ID 0x0444D01 (APP 1)

Fig 18 shows the Application ID 0x0444D01 (Delhi Metro app 1). Fig 18 also shows that it contains 10 (3)DES Keys and different permissions associated with it.

Fig 19 shows Application ID 0x444D02 (Delhi Metro app 2). Fig 19 also shows that it contains 5 (3)DES Keys and different permissions associated with it.

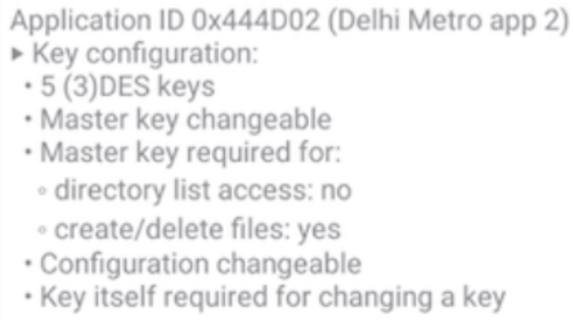

Fig 19. Application ID 0x444D02 (APP 2)

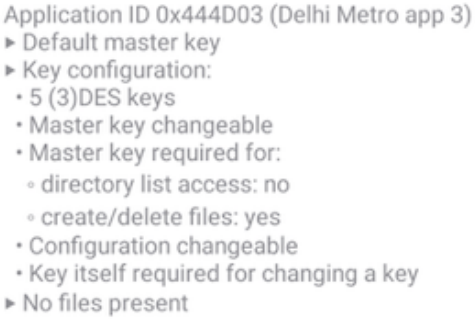

Fig 20. Application ID 0x0444D03 (APP 3)

Fig 20 shows Application ID 0x444D03 (Delhi Metro app 3). Fig 20 also shows that it contains 5 (3)DES Keys and different permissions associated with it.

Fig 21 shows Application ID 0x444D04 (Delhi Metro app 4). Fig 21 also shows that it contains 5 (3)DES Keys and different permissions associated with it.

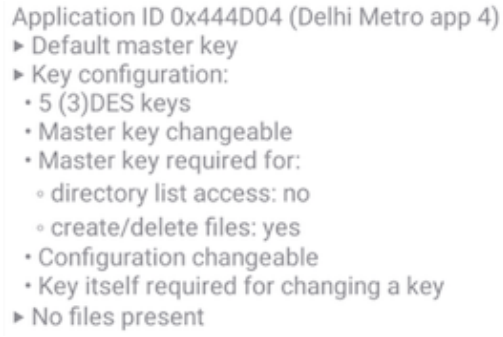

Fig 21. Application ID 0x444D04 (APP 4)

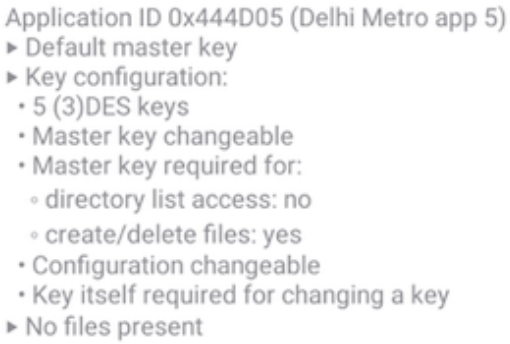

Fig 22. Application ID 0x444D05 (APP 5)

Fig 22 shows Application ID 0x444D05 (Delhi Metro app 5). Fig 22 also shows that it contains 5 (3)DES Keys and different permissions associated with it.

Fig 23 shows Application ID 0x444D06 (Delhi Metro app 6). Fig 23 also shows that it contains 5 (3)DES Keys and different permissions associated with it.

DOI: 10.5281/zenodo.5171426

Received December 30, 2020 Accepted Jun 01, 2021

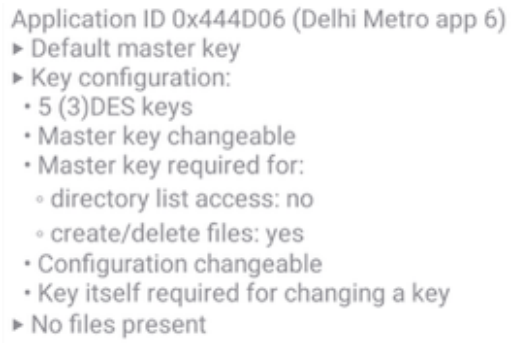

Fig 23. Application ID 0x444D06 (APP 6)

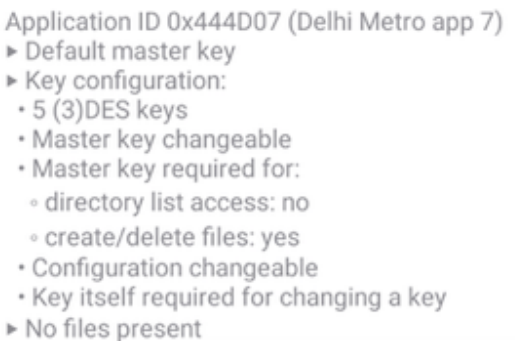

Fig 24. Application ID 0x444D07 (APP 7)

Fig 24 shows Application ID 0x444D07 (Delhi Metro app 7). Fig 24 also shows that it contains 5 (3)DES Keys and different permissions associated with it.

5. Result and Conclusions

We can successfully read, dump and write the data of the MIFARE Classic Card and Token using RC522 Module and Arduino Mega. We can read the data structure of the Delhi metro card. Delhi Metro Card is a MIFARE DESFire card and has a different data/file structure from MIFARE Classic Tag. Delhi Metro card has a 7-Byte Unique Identifier (UID), whereas MIFARE Classic Card & Token have only 4-Byte Non-Unique Identifier (NUID). Delhi Metro card uses (3)DES to encrypt the data in it. Delhi Metro Card has a total size of 4kB out of which 2.5kB is available. MIFARE Classic Card & Token are only 1kB in size. Delhi Metro Card has 7 different app sections, each section has separate keys which gives the read/write permission for that section.

Table III summarises all the findings and important characteristics in a tabular format.

Table III RESULTS

	MIFARE CLASSIC CARD	**MIFARE DESFIRE CARD**
Size	1 KB	4 KB
ENCRYPTION	No	Yes, 3(DES)
UID/NUID	4 BYTE NUID	7 BYTE UID

6. Future Scope

Dumping the raw bytes of the Delhi Metro Card and reverse engineering the protocols by which the Delhi Metro Card holds, transmits the data & pass various checks at the Metro Station.

References

[1] https://en.wikipedia.org/wiki/Radio-frequency/_identification
[2] Jechlitschek, Christoph. (2010). A survey paper on Radio Frequency Identification (RFID) trends.
[3] Mohd Saad, Norhashimah. (2013). Barcode Recognition System. International Journal of Emerging Trends & Technology in Computer Science(IJETTCS) 2278-6856. 2. 1-6.
[4] Chang, Jae. (2014). An introduction to using QR codes in scholarly journals. Science Editing. 1. 113-117. 10.6087/kcse.2014.1.113.
[5] https://www.behance.net/gallery/67918923/RFID-Tag

DOI: 10.5281/zenodo.5171426

Received December 30, 2020 Accepted Jun 01, 2021

[6] https://rfid4u.com/dig-deep-construction-of-rfid-tags/
[7] www.makeuseof.com/tag/technology-explained-how-do-rfid-tags-work/
[8] Daniel M. Dobkin, The RF in RFID: Passive UHF RFID In Practice, Newnes 2008 ISBN 978-0-7506-8209-1, chapter 8
[9] Sen, Dipankar; Sen, Prosenjit; Das, Anand M. (2009), RFID For Energy and Utility Industries, PennWell, ISBN 978-1-59370-105-5, pp. 1-48
[10] https://www.nxp.com/docs/en/data-sheet/MF0ICU2.pdf
[11] https://www.iso.org/standard/73597.htm
[12] https://www.iso.org/standard/70171.html
[13] Patel, Sagar Talati, Prachi Gandhi, Saniya. (2019). Design of I2C Protocol.
[14] S. Saha, M. A. Rahman and A. Thakur, "Design and implementation of SPI bus protocol with Built-in-self-test capability over FPGA," 2014 International Conference on Electrical Engineering and Information Communication Technology, Dhaka, Bangladesh, 2014, pp. 1-6, doi:10.1109/ICEEICT.2014.6919076.
[15] Dakua, BiswajitHossain, MdAhmed, Foisal. (2015). Design and Implementation of UART Serial Communication Module Based on FPGA
[16] https://github.com/rishab-rb/RFID/blob/main/get%20UID
[17] https://github.com/rishab-rb/RFID/blob/main/Dump%20Info
[18] https://www.mifare.net/en/products/tools/nfc-taginfo-app

DOI: 10.5281/zenodo.5171426

Received December 30, 2020 Accepted Jun 01, 2021