The MIFARE Hack

Mathias Morbitzer
m.morbitzer@student.ru.nl

Radboud University Nijmegen

Abstract. The MIFARE Classic is the most popular RFID chip, used in public transport as well as cafeterias and other applications. This paper gives an overview of which methods were used to re-engineer the chips, from polishing to analyzing the communication. Within this process, various security weaknesses were discovered. Those allow various attacks, which will be shown in the last part of the paper.

Keywords: MIFARE, NXP, re-engineering, RFID, CRYPTO1, security

1 Introduction

The use of RFID cards gets more and more common. In general, two types of RFID cards are used, active and passive cards. While active systems have an own source of energy, passive systems rely on the energy provided by the cardreaders.

One of the most common RFID Cards is the MIFARE Classic, produced by NXP Semiconductors, which was formerly a division of Philips. Their headquarter is located in Eindhoven in the Netherlands. With a market-share of 75% of all transit cards and more than 650 cities using transport cards based on their MIFARE technology, NXP is clearly the market-leader in this technology. [2]

The MIFARE Classic card family, which covers not only the public transport, but also access management, loyalty cards and many more offers different types of cards, which are all passive and fully compliant with ISO/IEC 14443 Type A. [19]

For cryptographic purposes, the MIFARE Classic cards use the CRYPTO1 cipher. This is a proprietary encryption algorithm which was developed by NXP. The algorithm is based on the Hitag2 algorithm, which is used in HITAG RFID systems. The details of the algorithm are kept secret by the manufacturer. Such an approach is called "security through obscurity", and is not a good way to implement security [15]. Due trying to keep the algorithm secret, the hackers Nohl and Plötz from the Chaos Computer Club (CCC) in Berlin were able to discover the algorithm by reverse engineering (re-engineering). [14]

This paper shows which steps they used to re-engineer the algorithm and which attacks are possible now that the algorithm and its weaknesses are known to the public.

2 MIFARE

There are two common types of the MIFARE Classic cards, the MIFARE Classic 1k and 4k. The 1k Chip has 1k EEPROM memory, which is separated in 16 sectors with 4 blocks, each containing 16 byte. This makes a total of 64 blocks. The 4k version offers 4k of EEPROM memory, separated in 256 blocks, where 32 sectors have 4 blocks and additional 8 sectors having 16 blocks, as shown in figure 1. [16]

The first block contains the unique identification number (UID) of the card as well as some vendor specific data which is write protected. On the last block of each sector, access keys and access rules can be found. This block is not intended to store user data. [16][17] Before any memory operations are allowed on a sector, a reader has to authenticate itself for this sector. Therefore, the sector trailer is used, containing the secret keys A and B which are used for the authentication. The access conditions, which are located in the same block, specify which memory operations are allowed on the sector. [4][17]

For the sector trailer itself, there are specific access conditions. As mentioned, the trailer contains two keys, A and B. While key A is never readable, key B can be configured readable or non-readable. If it is declared as readable, only key A can be used for authentication and key B will be used to store data. There is also the single byte U, which has no defined purpose. [4][16]

The data blocks are used to store information for the application. On a cafeteria card for example, this could be the name of the owner plus the current balance of the card.

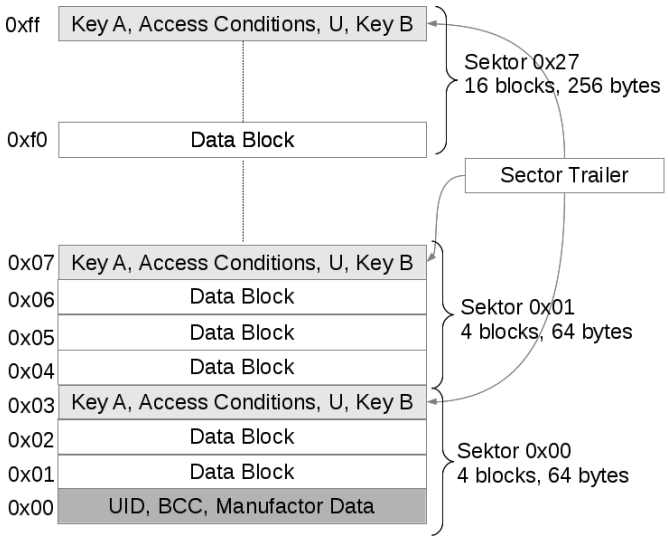

Fig. 1. MIFARE Classic 4k Memory (according to [4])

3 Re-engineering the MIFARE Classic

3.1 Choosing the card and the method[1]

Around 2006, the "Chaos Computer Club" (CCC) in Berlin, Germany, started to investigate the MIFARE Classic cards. The reasons for choosing this particular type of cards were obvious: For once, the card uses a proprietary algorithm for cryptographic purposes, although this kind of approach already turned out not to be secure in the past. The reason therefore is that those proprietary algorithms mostly do not have to undergo a check from third parties outside the producing company to detect flaws.

Another motivation factor was that the chip could not be secure because of the encryption key, which is only 48 bit long. That might have been enough in 1994, when the chip was released, but already in 1999, DES was cracked, which uses a 56 bit key.

The last reason for choosing those specific chips was that it was and is the RFID chip family with the highest market share [2]. Compared to other manufacturers, MIFARE tags and readers are also easily available, which alleviates the investigation.

Before the CCC started re-engineering, the most efficient known attack was a simple brute force attack. As the algorithm was not known, this had to be done directly against the card. The card itself does not memorize a value like the amount of fail attempts in the last minute. This is because the power of the card is only supplied by the reader and writing in the EEPROM would take too much time.

The main problem when executing such an attack in this scenario is the duration. Trying one attack key takes about 5ms. Taking into account the 48 bit key, it would take up to 55,000 years until the right key is found. Transferring the whole process to a computer would decrease the duration massively, but this was not possible due the algorithm was not known to the public.

For these reasons, it was necessary to re-engineer the algorithm. Already before, algorithms have been revealed through disassembling. Examples therefore are A5/1 and A5/2 algorithms, responsible for securing GSM communication [1], and the Hitag2 and Keeloq algorithms, which are used in remote controls for cars [3]. But those were all software implementation, which could be re-engineered out of firmware, binaries or other software. For the MIFARE chips, there is no software implementation. All calculations are done in hardware.

Another option would have been a blackbox analysis. Using this method, various inputs are sent to the card, and the output is analyzed. There has also been successful re-engineering via blackbox analysis before, but the problem for the MIFARE was the 48bit cipher, which would have made it too hard to guess correct which operations were used from the algorithm.

The next problem the CCC was confronted with was that the card does not answer if the wrong key is used. According to Henryk Plötz, this is one of the

[1] This chapter is based on [17]

few things which have been designed well in the MIFARE Classic cards. Using this system, the reader first has to authenticate itself to the card, showing that it knows the key. Otherwise, if the card authenticates itself first to the reader, it would have to send data to it. If the reader does not know the key, it would get valuable information from the card this way.

Summed up, the analysis of software was not possible during non existence, and a blackbox analysis would require too much luck. But there was still the hardware, which executes the algorithm. So the team decided to directly analyze the hardware.

3.2 From the Hardware to the algorithm[2]

The process of re-engineering started with a MIFARE Classic card which was used in a cafeteria. It was a booking card, that means its only purpose was to charge balance and pay with that balance.

Before the first step, it is important to mention that a RFID card is not the same as an RFID chip. The card consists of a tiny chip, which is surrounded by plastic to extend it to credit card size.

So firstly, the CCC needed to get the chip out of the plastic. To remove the plastic, there are two possibilities. One of them is to use acetone. As alternative, red fuming nitric acid can be used. But therefore, a laboratory is needed as it is complicated to handle. To keep the complexity low, the solution with acetone was chosen. To skip this step, it is also possible to directly buy just the chips. This is possible to provide companies the option to build their own cards.

After the plastic is dissolved, the silicon chip with the size of about 1x1mm is left. Such chips consist of various layers. In the case of MIFARE Classic, the chips have seven layers. For the analyze of those layers, a standard microscope with a camera, which sends the recorded images directly to a computer, was used. The matter of expense of such a microscope is about €1000.

Next, the re-engineers were confronted with the problem that only the first layer of the chip is visible. The other layers might be slightly shining through the first one, but this is not enough to perform a full analysis. So in the second step, a way to analyze all layers of the chip was needed. One option therefore was chemical etching. As alternative, the chip could be mechanically polished, which was the way chosen by the CCC. Like in the step before, the reason was to keep the complexity as low as possible. For the polishing, polishing emulsion and sandpaper with a very fine grading of 0.04μm were used. [14]

In this step, the problem was that it is necessary to polish the chip equally in order to have a good view of all layers. The chip itself is not even because of wires for antennas and other connections. Also, with the size of about $1m^2$, it is hard to be handled with bare hands. Therefore, the CCC decided to put plastic mass around the chip. With this, they did not only had an object which was easier to handle, but also easier to be polished layer by layer as they could make the block easily even on one side.

[2] This chapter is based on [11] and [17]

The polishing process is very slow and has to be done very carefully. There is no possibility to restore anything which has been polished of the chip before. Each of the layers which needs to be analyzed is about 1μm thick. In total, the team wasted about 10 chips before they had all the pictures they needed. Henryk Plötz guessed that this whole process, including the learning procedure, would take about 2 weeks of work. But this number was only estimated, as the work was done in leisure time and not on a professional level.

After the CCC made pictures of all layers, the next step was analyzing and reconstructing the logic out of them. In total, there were about 10,000 gates to be analyzed. Fortunately, those gates only consisted of about 70 different types. To accelerate the analysis, the team created a MATLAB[3] script which needs one gate as input, and afterwards finds gates everywhere else on the chip, even if they are rotated. This script is meanwhile released under the name "degate" [18]. At the end, this process results in an overview of the chip, showing in which place which type of gate is used. Additionally, all discovered gates and pictures of them were listed on the website "The Silicon Zoo" [13]. Figure 2 shows how the images of the chip look like before and after the use of "degate". [14].

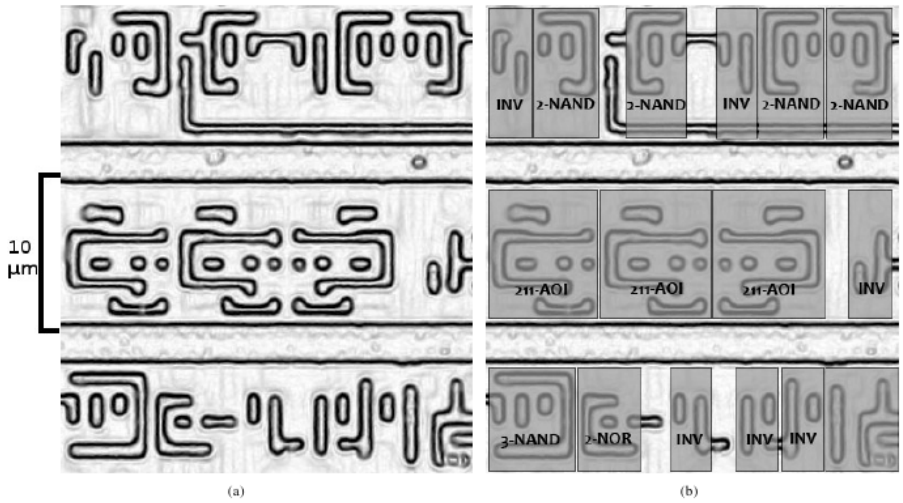

Fig. 2. (a) Source image of the second layer; (b) after using "degate"[14]

In the next step, the team manually followed all the connections between the gates. On each layer, there are about 100 horizontal and 100 vertical connections. Because of the huge amount, it would be too much effort to analyze the whole chip. So the team focused on the crypto logic, which forms only 10% of the chip. To identify which part of the chip is responsible for the crypto logic,

[3] More information about MATLAB: http://www.mathworks.com/products/matlab/

the map of gates created in the previous step was very useful, because it was possible to analyze groupings of gates. For example, XOR- and flip-flop gates are basically only used for cryptography. If the chip than has a place which shows an accumulation of such gates, there is a high chance that this forms the crypto logic. When then 48 XOR-gates where found next to each other, this was a sign that the team had found the right place on the chip, counting the fact that the key of the MIFARE Classic is also 48 bit long.

The crypto logic consists of about 1000 gates and 1500 connections. For the analysis of those, the team followed each connection, wrote down which one ends where and which input relates to which output. This process is very time-consuming, as for example connections sometimes cross up to three different layers or can also run all over the chip. Therefore, it is also very error-prone and takes a lot of time. In total, this step of the re-engineering process took again about 2 weeks.

This last step of re-engineering created a circuit diagram, out of which it was possible to create a block diagram, which showed the encryption process. So, after investing two years of leisure time, the cryptographic algorithm of the MIFARE Classic cards was re-engineered. It will be explained in detail in section 3.4.

3.3 The initialization process

Now that the algorithm of the MIFARE Classic cards was re-engineered, the only thing missing was the initialization process and therefore the initial state of the algorithm, due it is not part of the crypto logic. The process was identified with testing the radio protocol, which will be discussed in this section.

For the testing of the initialization process, the CCC used OpenPCD as a reader. This is a free programmable RFID Reader, which allows good control over the data being sent and received. [8]

Other researchers from the "Radboud University Nijmegen" in the Netherlands used a Proxmark III [5] to eavesdrop the communication between the reader and the card. This tool was developed by Jonathan Westhues.

As mentioned before, the card does not give an answer if the wrong key is used for authorization. This is a good design of the card, which resulted in more work for the re-engineers. Therefore, in the first phase of the process to discover the initialization process, the correct key was sent to the card. Afterwards, the data being sent was altered bit by bit to check if there is still a response. With switching various bits, it was discovered that the UID and the key are used as input for the initialization. To the CCC, this discovery was no big news, as it is also mentioned in the MIFARE documentation which they had at their disposal.

From the re-engineering of the chip, it was known that the crypto logic had only one input. But because they also knew that there was the UID used as input as well as the key, those two had to be connected somehow. Probably the connection method was XOR, but that was only a suspicion. Trying to prove this theory, the first bit of the UID was flipped as well as the first bit of the key. If they were connected via XOR, this should still result in the same value. As

confirmation for the team, the card responded again, which meant the theory of the connection via XOR was correct. With testing, they found out that the first five bit of the UID and the key are connected directly, which means the n^{th} bit of the UID is connected to the n^{th} bit of the key. For the other bits, the relationship is different. More information about these relationships can be found in [16].

Another discovery made during testing the radio protocol was that the algorithm uses a pseudo random number generator (PRNG), which is realized with a linear feedback shift register (LFSR). Along general lines, a LFSR creates a sequence of numbers which look different from each other. It always starts with the same number and moves to the next number with the same frequency. Also, the n^{th} number of one LFSR will always be the same. More information about LFSR can be found at [9].

A problem that is created with using LFSR in passive RFID tags is that the cards rely on power supply from the reader. This means, every time the card is removed from the reader, it is out of power and the LFSR starts again with its initialization sequence as soon as it is provided again with energy, sending the same numbers in the same frequency as the last time. This was proved in an experiment of the CCC. They put the card next to the reader, and then switched on the reader. As a result, it always needs approximately the same time after the card is supplied with energy and starts creating random numbers, that the reader starts communicating. Within this experiment, they had the same random values of the LFSR in 12 out of 27 attempts. [16]

Gans et al. acknowledged this weakness. They state they were able to produce about 600,000 random values per hour, in which a nonce was repeated with a minimum of four times. This means that in theory a random values could reappear after 0.618 seconds [4]. So, the first possible attack of a MIFARE Classic card was discovered, the replay attack. A closer explanation of this attack can be found in section 4.2.

3.4 The algorithm

With the work done by the CCC, it is now possible to present the CRYPTO1 algorithm of the MIFARE Classic cards in detail. As shown in figure 3, the algorithm consists of a LFSR and a function f(x). [12]

In the beginning, the shift register is initialized with the secret key. Afterwards, the PRNG creates 32 bit and XORs them with the UID. This string will then be shifted into the state of the shift register [16]. The created random value is also used for the challenge response protocol between reader and card [10].

At this point, the encryption is activated and all incoming and outgoing bits are XORed with the keystream. The shiftregister itself is only used for bits containing data or cyclic redundancy check (CRC) values. For encrypting parity bits, a bit of the keystream which was already used for a data bit is used again. [16]

Each cycle, the function f(x) computes one bit for the key stream out of 20 bits it gets from the LFSR. The 18 taps of the LFSR are used to fill the first

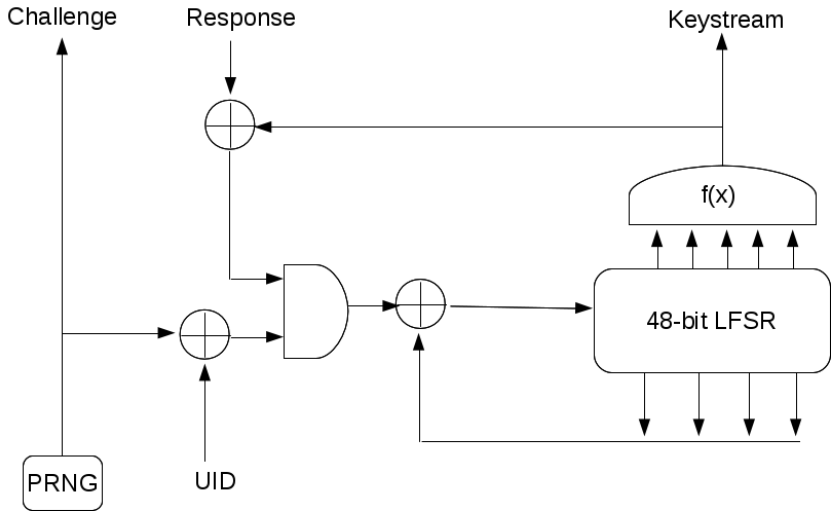

Fig. 3. Crypto-1 stream cipher and initialization values (according to [12])

register bit on each shift. For this, the taps are connected linearly. Because of this linear connection, the update function does not contain any non-linearity, which means that when one of the shift values is given, it is possible for an attacker to calculate previous values as well as upcoming values. With today's knowledge of crypto systems, this can be considered a serious weakness. [14][17]

4 Possible Attacks

This section shows attacks which are possible now that the CRYPTO1 cipher, the initialization and the weak PRNG are known to the public. Besides the attacks mentioned, there are also other possibilities, like for example privilege escalation, keystream shifting as well as various card-only attacks.

4.1 Brute Force

Before the CRYPTO1 algorithm was known to the public, the best attack was a simple brute force against the card. Among others, this is possible because the card does not safe a state like the amount of wrong authentication attempts. For once, because the card has no own energy supply, this would require a writing process on the chip, which takes too much time. Another reason for not remembering the number of failed login attempts and locking the card after a certain amount is that this would be another attacking possibility. An attacker could execute a Denial of Service (DoS) attack by just standing next to somebody with a RFID card, trying to authenticate with random keys as long as the card locks.

As the brute force could only be done directly against the card, trying one single key would take about 6ms. The key has 48 bit. This means trying each possible combination would take approximately 55.000 years. [17]

After the algorithm is known to the public, the dutch organization TNO guessed that building a machine to crack the MIFARE Classic key in two hours would cost about $9000 [22]. According to the CCC, this estimation is too much, a normal PC should be enough to crack the key in acceptable time. [17]

4.2 Replaying

The replay attack is based on the weak PRNG used in the MIFARE Classic cards. As soon as the card is held next to a reader, it receives power and starts generating random values. The problem here is that the LFSR always generates the same values with the same frequency, which means at time t_n after the card receives power, the nonce which is generated by the PRNG will always be the same. [16][17]

If an attacker now eavesdrops a communication between a reader and a tag, it is possible to calculate at which time t_n after activation the tag started the communication. When the tag is in possession of the attacker, it is not even necessary to have the exact time. A high possibility for guessing the right time would be enough. [16]

The attacker can now replay the transaction at the same time t_n after the card receives power, and will get the same nonce. As alternative, it would also be possible to change the transaction. This is possible because no integrity check is performed. [16][17]

There are various scenarios where this could be interesting. As for example, when the communication eavesdropped increases the balance of a booking card, like "add €5 to the current balance of the card". This transaction could then be replayed as often as desired to increase the balance.

Another possible scenario would be to replay an authentication, as shown in [4].

4.3 The "Ghost" device

The "Radboud University Nijmegen" in the Netherlands developed a device called "Ghost", which is a programmable RFID tag. Like a normal tag, it is able to communicate with a reader. But different to a normal tag, the microchip which is used on the device can be programmed. Verdult also wrote a firmware allowing to control the complete communication bytes which are sent between reader and tag. The device uses a RS232 [21] interface with which it is able to communicate with the serial port of a computer. [6][23]

With "Ghost", it is possible to clone an RFID tag. First, the user can configure a UID. Afterwards, some received bytes and the following bytes being sent can be defined by the user to set the reaction of the device when certain frames are received from the reader. According to Verdult, this might be for example

useful when a reader only wants to identify the tag and therefore requests one simple answer which is not encrypted. [23]

After configuration of the device, it can be disconnected from the computer and used as a standalone device. This can be used for a replay attack. If communication between a reader and a tag is known to the attacker, he can use the '"Ghost" to clone it. [23]

Besides that, it is also possible to accomplish man in the middle (MITM) attacks with the device. Therefore, the original tag and reader, the OpenPCD reader and the "Ghost" connected to a computer are needed. What the device then does is communicating with the original reader and forwarding the requests to the computer. There, the requests are being processed through the OpenPCD to the original tag. When the tag answers, it answers the OpenPCD reader, which sends the data to the computer. This data is forwarded to the "Ghost", which sends them back to the original reader. The scenario is shown in figure 4. [23]

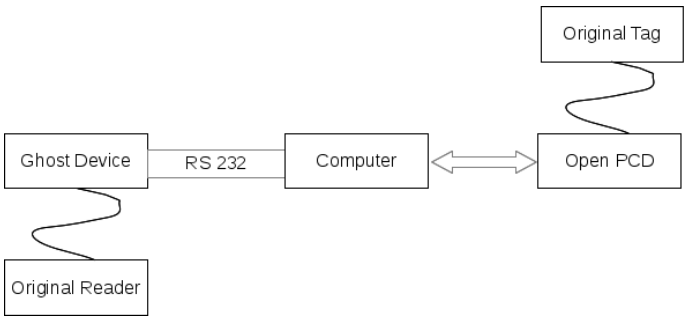

Fig. 4. Man in the middle scenario (according to [23])

In the described scenario, it is possible to manipulate the data being sent from the original reader to the original tag. One reason therefore is that the integrity of the received data is neither checked by the reader nor the tag. [17]

4.4 Keystream recovery

Another attack has been shown by Gans, Hoepmann and Garcia [4]. This one is also based on the weak PRNG. Gans et al. used the Proxmark III to create the same nonce multiple times, which automatically also leads to the same keystream. Because the same keystream is connected twice with the data via XOR, it is possible to read one of the cleartexts connected with the keystream if the other cleartext is known. [4]

If an attacker records a transaction which includes a reading command, he can replay this transaction, but change the block that should be read. To get a

known cleartext, a special property of the access rights of the MIFARE Classic cards is used: Key A can never be read. If somebody tries to read it, the card will return 0's. When key B is used as key and not to store data (which is mostly the case), the same holds for key B. [4][17]

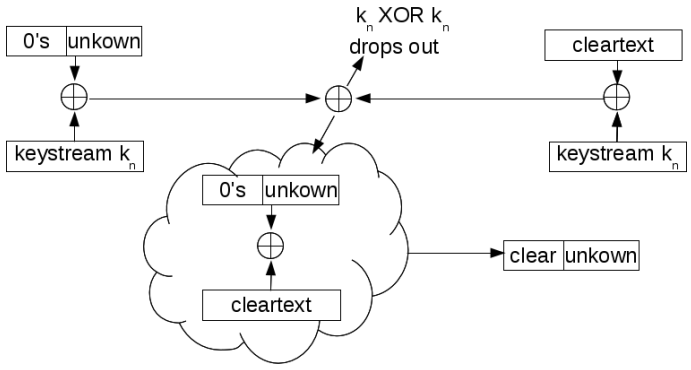

Fig. 5. Keystream recovery

So the card returns 6 Bytes of 0's, XORed with the keystream k_n, when the operation is done at time t_n. When an attacker now reads another block also at time t_n after the card is supplied with power, this block is encrypted with the same keystream k_n and afterwards sent by the card. If the two messages received from the card are now XORed, this drops out the keystream k_n as XORing twice with the same values results in the start value. What is left are 6 byte of 0's and some unknown text, which are XORed with the cleartext of the second block. Now that the first message is known to contain 6 byte of 0's at the beginning, the first 6 byte cleartext of the second block can be read, as shown in figure 5. [4][17]

4.5 Recovering the secret key

Garcia et al. provided two attacks in their paper, with which they are able to calculate the secret key of the MIFARE Classic cards within one day and less. [7]

For their first attack, they are splitting the 48 bit of the key in an online and an offline search. At first, offline, they create a table, containing states of the LFSR and the first 64 byte of the related keystream. Such a table can be computed within one afternoon on standard hardware and needs approximately one terabyte of space. For repeating the attack, the table can be used again. [6][7]

When the table is created, the group creates 2^{12} authentication sessions with the reader for the online part. For each of these sessions, they compute the next

64 bit of keystream. This does not require access to a tag, only to a reader. If now the same keystream is used in the online and the offline part, it is possible to look up the internal state of the cipher. To afterwards compute the secret key, which is equal to the initial state of the LFSR, it is enough to run the cipher backwards to reach this initial state. [7]

Garcia et. al claim that within one second, between 5 and 35 authentication sessions with the reader can be recorded, depending if it is an online or an offline reader. Therefore, gathering all the 2^{12} authentication sessions needed for the attack will take between 2 and 14 minutes where the attacker is required to dwell next to the reader. [7]

The other attack developed by Garcia et. al takes even less time and can also be found in [7]. This attack relies on the fact that the team is able to invert the function f(x) of the CRYPTO1 algorithm. This invertibility allows them to recover the state of the LFSR at the end of an authentication. Afterwards, similar to the first attack, they calculate back to the initial state. Using this attack, they only need one single authentication session, compared to 2^{12} authentication sessions with the attack mentioned before. [7]

An implementation of this attack needs, according to the team, less than one second and only a few megabyte of RAM to recover the secret key. On contrary to the first attack in this section, this one also does not need any precomputing work. The implementation can even be optimized which drops the time needed to reveal the secret key to under 0.1s, using the same hardware.

Nohl also developed a method to recover the secret key within a minute, which he explains in [12]. Similar to the other attacks to recover the secret key, this one is also based on attacking the LFSR.

5 Consequences

When the first vulnerabilities were known to the public, NXP commissioned the Dutch organization TNO to counter-prove those. In the following report from the TNO, the vulnerabilities discovered were acknowledged, but the organization did not see a big risk in them. [22]

After more and more vulnerabilities and attacks were released, NXP decided to release their new RFID tag, the MIFARE Plus. As suggested by the CCC [14], the new card does not use a proprietary encryption cipher anymore, but uses the Advanced Encryption Standard (AES). Besides that, the new tag also supports MIFARE Classic. This offers the possibility to slowly migrate from the old tags to the new MIFARE Plus tags. [17]

On their website, NXP also published a list of their activities to keep their RFID cards safe. This includes among others a court case, trying to prohibit the Dutch group from publishing their findings, which is described closer in [6]. The list can be found at [20].

Meanwhile, a lot of companies exchanged their MIFARE Classic cards with MIFARE DESFire, which provides better hardware as well as better security

features than the MIFARE Classic cards. In transports, the Dutch OV Chipkaart has been exchanged with a card which provides a better PRNG as well as protection against side channel attacks. Similar to other cards, like the London Oyster card, used for public transport in London, upgrading to different systems is planned, but still in progress. [17]

To improve future developments, the foundation openticketing.eu has been established, and the Dutch ministry is advancing the use of open cryptographic designs and communication standards as well as a closer collaboration of companies with academia. [6]

Once again, "security through obscurity" has been proved not to be feasible. Rather then keeping an cryptographic algorithm secure by keeping it secret, it is better to make use of public cryptographic standards like AES which are known, tested and accepted by the public.

References

1. Ross Anderson. A5. Posted on sci.crypt, June 1994.
2. Dan Balaban. Vendor Group Seeks to Crack Mifare Dominance. http://www.nfctimes.com/report/vendor-group-seeks-crack-mifare-dominance, 2010. [Online; Request on March, 12th of 2012].
3. Andrey Bogdanov. Attacks on the KeeLoq Block Cipher and Authentication Systems. In *In RFIDSec*, 2007.
4. Gerhard de Koning Gans, Jaap-Henk Hoepman, and Flavio D. Garcia. A practical attack on the mifare classic. In *CARDIS*, pages 267–282, 2008.
5. Gerhard de Koning Gans and Roel Verdult. Proxmark.org. http://www.proxmark.org/, 2012. [Online; Request on April, 4th of 2012].
6. Flavio D. Garcia. The Fall of a Tiny Star. http://www.cs.ru.nl/ flaviog/publications/Tiny.Star.pdf, April 2012.
7. Flavio D. Garcia, Gerhard Koning Gans, Ruben Muijrers, Peter Rossum, Roel Verdult, Ronny Wichers Schreur, and Bart Jacobs. Dismantling mifare classic. In *Proceedings of the 13th European Symposium on Research in Computer Security: Computer Security*, ESORICS '08, pages 97–114, Berlin, Heidelberg, 2008. Springer-Verlag.
8. Bitmanufaktur GmbH. OpenPCD Passive RFID Project - OpenPCD. http://www.openpcd.org/, 2012. [Online; Request on April, 2nd of 2012].
9. Texas Instruments. What's an LFSR? http://www.ti.com/general/docs/lit/getliterature.tsp?literatureNumber=scta036a&fileType=pdf&track=no, December 1996. [Online; Request on April, 2nd of 2012].
10. Mechanisms using symmetric encipherment algorithms, 1999.
11. Karsten Nohl. 25C3 Talk: Hardware Reverse Engineering. http://video.google.com/googleplayer.swf?docid=-1443778510407719537&hl=en&fs=true, December 2008. [Online; Request on March, 12th of 2012].
12. Karsten Nohl. Cryptanalysis of Crypto-1. http://www.cs.virginia.edu/ kn5f/ Mifare.Cryptanalysis.htm, 2012. [Online; Request on April, 4th of 2012].
13. Karsten Nohl. The Silicon Zoo. http://siliconzoo.org/, 2012. [Online; Request on March, 15th of 2012].

14. Karsten Nohl, David Evans, Starbug Starbug, and Henryk Plötz. Reverse-engineering a cryptographic RFID tag. In *Proceedings of the 17th conference on Security symposium*, pages 185–193, Berkeley, CA, USA, 2008. USENIX Association.
15. Bruce Perens. Why Security-Through-Obscurity Won't Work. http://slashdot.org/features/980720/0819202.shtml, 1998. [Online; Request on March, 12th of 2012].
16. Henryk Plötz. Mifare Classic - Eine Analyse der Implementierung. Diploma thesis, Institut für Informatik, Mathematisch-Naturwissenschaftliche Fakultät II, Humboldt-Universität zu Berlin, 2008.
17. Tim Pritlove. Interview with Henryk Plötz. Radio Interview in CRE098, September 2008.
18. Martin Schobert. Reverse engineering integrated circuits with degate. http://www.degate.org/, 2012. [Online; Request on March, 15th of 2012].
19. NXP Semiconductors. MIFARE Classic. http://www.nxp.com/products /identification_and_security/smart_card_ics/mifare_smart_card_ics/mifare_classic/, 2012. [Online; Request on March, 12th of 2012].
20. NXP Semiconductors. Security of MIFARE Classic. http://mifare.net/technology/security/mifare-classic/, 2012. [Online; Request on April, 7th of 2012].
21. Christopher E. Strangio. The RS232 Standard. http://www.camiresearch.com/Data_Com_Basics/RS232_standard.html, 2012. [Online; Request on April, 7th of 2012].
22. Security Analysis of the Dutch OV-Chipkaart, 2008.
23. Roel Verdult. Security analysis of RFID tags. Master thesis, Radboud University Nijmegen, 2008.
24. Roel Verdult and Flavio D. Garcia. Meeting regarding MIFARE Classic. Meeting, April 2012.