International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 6, Issue 10, October 2016)

Cryptanalysis and Improving Variety of Weaknesses on MIFARE Classic

M. M. Abdelwahab[1], M. R. M. Rizk[2], El-S. A-M. El-Badawy[3], H. A. Selim[4]
[1,2,3]*Electrical Engineering Department, Alexandria University, Faculty of Engineering, Alexandria, Egypt*
[4]*Computer Science Department, Arab Academy for Science and Technology, Alexandria, Egypt*

***Abstract*— Mifare Classic contactless smart card has a ubiqtous nature that, it covers about 85% of the contactless smart card market because of its low price, handy implementation, and no required maintance. It implements a proprietary symmetric-key mutual authentication protocol and a proprietary stream cipher algorithm known as CRYPTO1, both of which have been reverse engineered and many weaknesses were found and exploited by different attacks. The existing attacks in various scenarios prove that Mifare Classic does not provide the expected 48-bit security level. The most practical scenario is the card-only scenario where a fake, emulated reader has an access to a genuine card. This paper presents an extensive collection of all weaknesses affecting Mifare Classic card, then a broad comprehensive survey of all known attacks on Mifare Classic is introduced in a simple, conceptually manner with great details, puting them into the right place according to the known art of cryptanalysis. We also proposes number of solutions to overcome variety of weaknesses that will strengthen and rescue its degraded security level easily, consequently again Mifare Classic becomes trustworthy.**

***Keywords*—Attacks, Mifare Classic, Modification, Solutions, Weaknesses.**

I. INTRODUCTION

The Mifare Classic smart card of NXP Semiconductors is one of the most widely used contactless card in many application such as access control systems and public transportation. Although internal structure and ISO-14443-A communication protocol of the Mifare Classic are proprietary, it could not prevent researchers from reverse engineering it and discovering its weaknesses. The Mifare Classic chip is a memory chip having some additional features with two types; which are1k and 4k. Size and organization of the memory are the difference between both types. The memory in the 1k card is organized into 16 sectors of 4 blocks each, whereas, in the 4k card, the first 32 sectors consist of 4 blocks, but the remaining 8 sectors are 16-block sectors. The reader should be authenticated successfully for the card sector before any memory operations on the data blocks of that sector are permitted, however the last block of each sector which controls the authentication, is known as the sector trailer.

The Mifare Classic implements a proprietary symmetric-key mutual authentication protocol and a proprietary encryption algorithm known as CRYPTO 1 which is a stream cipher uses 48-bit secret key to encrypt messages between card and reader.

A. Crypto 1 Stream Cipher

Crypto1 stream cipher utilizes a 48-bit linear feedback shift register (LFSR) and a two-layer nonlinear filter generator, both of them form the pseudo-random bit generator [1] as shown in figure 1 where, the 20 output odd bits of LFSR starting from the bit number 9 and ending with bit number 47 are fed to the five 4-bit functions of the first layer which, in turn direct their outputs to a second layer that produces one keystream bit. This process is repeated each clock cycle after the LFSR shifts its state one bit to the left.

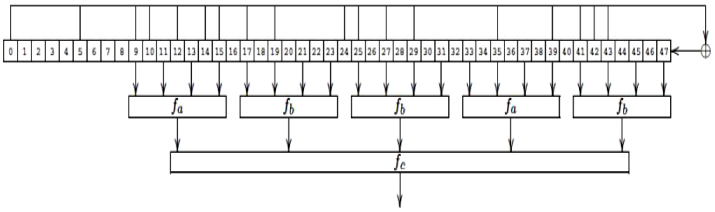

Figure 1: Keystream generation of CRYPTO 1 [1]

B. Mifare Classic Authentication Protocol

The Mifare Classic card performs a challenge- response mutual authentication with the reader, where the 32-bit challenge nonces n_T and n_R used by the tag and the reader are produced by the respective pseudorandom number generators. Although, the challenge is a 32-bit nonce that is generated by a pseudo-random number generator, it was discovered to be a 16-bit LFSR [1, 2] which is a separate circuit from that of the 48-bit LFSR used for the keystream generation. According to the standard ISO-14443-A, every plaintext byte sent is followed by a parity bit for error detection, where the ISO-14443-A standard specifies odd parity to be used that computed as the complement of the exclusive-or of all 8 bits in the byte.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 6, Issue 10, October 2016)

The parity bits in Mifare Classic communication protocol are computed over the plaintext instead of the ciphertext and each parity bit is then encrypted with the same keystream bit used to encrypt the next bit of the plaintext.

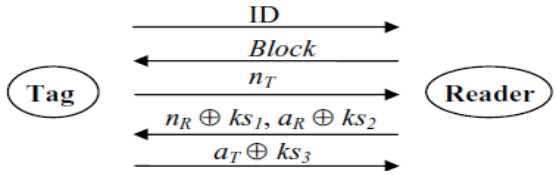

Figure 2: Authentication protocol of Mifare Classic [2]

The major steps of the three-pass mutual authentication protocol are illustrated in figure 2. In the introductory two steps, the tag sends its unique ID and the reader requests authentication for specific block, and then the next 3 steps occur as follows:

1. PASS 1: The tag sends a challenge nonce n_T to the reader.

2. PASS 2: Starting from this step, communication is encrypted where the reader replies with an encrypted reader nonce n_R using ks_1 which can be expressed as $\{n_R\}$ and an encrypted reader response a_R using ks_2 expressed by $\{a_R\}$,so $\{a_R\}$is the response to the challenge n_T.

3. PASS 3: If any of the eight parity bits of the preceding message is incorrect, the tag will not respond to the reader, and if all eight parity bits are correct, but the response $\{a_R\}$ is incorrect, the tag will respond with an encrypted 4-bit error code, otherwise these two conditions, the tag will respond with an encrypted tag answer a_T using ks_3 expressed by $\{a_T\}$.

It is worth pointing out that reader and tag responses, a_R and a_T ,are both derived from tag nonce n_T [1] according to the following equations:

$$a_R = \text{suc}^{64}(n_T) \qquad (1)$$
$$a_T = \text{suc}^{96}(n_T) \qquad (2)$$

Where, suc is the successor function that discards the leftmost bit and inserts a new rightmost bit generated using the feedback function.

In the following sections we explain the different weaknesses, and then we discuss the attacks on Mifare Classic. After that we propose solutions to improve security issues. Finally, we discuss the conclusions.

II. Weaknesses Of MIFARE Classic

Starting from late 2007, many researchers discovered the specifications and weaknesses of the Mifare Classic, where several weaknesses of the cipher have been found via reverse engineering and cryptanalysis [1, 3-5]. In general, the weaknesses can be classified under two main categories, which are weaknesses in the cipher design; or insufficient key space, where key size of 48 bits can be seen as susceptible to brute force attacks [6].

A. Weakness 1: Low Entropy of Random Generator

It is a weakness in the pseudo-random number generator, where the generator is 16-bits wide used to generate 32-bit sequence representing the nonce as shown in figure 3. This implies resulting an entropy of 65535 (2^{16}-1) if the LFSR state is assumed to be uniformly random. This is made worse because the LFSR starts from the same state after tag powering up, and shifting its state every 9.44 μs, so it repeats its state after about 618ms, thus the variable time before producing the tag nonce is the only randomness factor. Using x_n to denote the bit value (0 or 1) at position n and $\oplus$ is the exclusive-or logic function, its feedback function is defined by L ($x_{16}\, x_{17}\, \ldots\, x_{31}$) Where:

$$L\,(x_{16}\, x_{17}\, \ldots\, x_{31}) = x_{16} \oplus x_{18} \oplus x_{19} \oplus x_{21} \qquad (3)$$

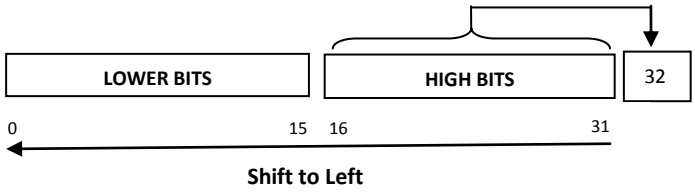

Figure 3: 32-bit generation of nonce using 16-bit LFSR

B. Weakness 2: Only Odd State Bits Used as Inputs to Filter Generator to Generate Keystream

It is a weakness in the cryptographic cipher. According to figure 1, we notice that the inputs to the non-linear function are the twenty odd-numbered bits beginning with bit number 9 to bit number 47, this allows us to use a Divide-and-Conquer technique, in which we split the even and odd bits into two groups, and at first focus on the odd group. The odd 9th to 47th bits are used to generate a keystream bit. Then, after two shifts to the LFSR, the 19 original bits in position 11 to 47 and a new bit in position 49 added to the right to form the new 20-bit input to the non-linear function, subsequently they used to generate the next keystream bit.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 6, Issue 10, October 2016)

These 21 bits (9,11, ..., 49) are in fact used to generate two consecutive even keystream bits. If we know the values of these two keystream bits, we can decrease the possibilities for the 21 bits by eliminating those that do not generate the right keystream bits. Similarly we can repeat the same process for the even group. Due to this weakness, the attacker can obtain a specific LFSR state after knowledge of some consecutive keystream bits which is in fact a serious weakness.

C. Weakness 3: First Nine Leftmost Bits of LFSR Not Used as Inputs to Filter Generator

Also, another weakness in the cryptographic cipher is described, in which the first nine bits from position 0 to 8 are not used at all as inputs to the filter function, this permits the adversary to perform an attack that is considered to be the reverse of the filter generation function known as rollback the LFSR state bit by bit. It is quite simple to repeat the rollback enough number of times to recover the initial state of the LFSR, which is the secret key. For instance, suppose we know the LFSR state immediately after n_R has been fed into the LFSR. We know that at first the LFSR is fed with ($u_{id} \oplus n_T$), where u_{id} is the 32-bit identification of the card and then subsequently with n_R. The rollback technique is run as follows:

1. We shift the LFSR to the right, that will drop the rightmost bit and we set the leftmost bit to an arbitrary value r.
2. We use the filter generator to compute the keystream bit used to encrypt the bit n_{R31}, since this leftmost bit r that we have just inserted does not participate in the filter generation, the choice of r does not affect the value of the keystream bit.
3. We exclusive-or the keystream bit that we have just obtained, with encrypted $n_{R31}$to recover n_{R31}.
4. We use n_{R31} to calculate the correct value of r by using the feedback function of the LFSR.
5. We repeat this procedure 31 more times to recover the LFSR state before n_R is shifted in, and 32 more times to recover the secret key before ($u_{id} \oplus n_T$) is shifted in. as a result to this weakness, the attacker can obtain the initial LFSR state (the secret key) after knowledge of a particular state with the card identification which seems to be a serious weakness.

D. Weakness 4: Statistical Bias in Cipher

According to many experiments of researchers [2], it is found that with probability about 0.8, the last 3 keystream bits of ks_1 are independent of the last 3 bits of $\{n_R\}$, where in fact the probability should ideally be 0.5.

Consequently, each nonzero 3-bit change (Δ_3) of last 3 bit of $\{n_R\}$ will cause the same change of the last 3 bits of n_R itself. Since the LFSR sequence depends linearly on n_R, this implies that for each value of Δ_3, the subsequent LFSR sequence will change linearly in a way that depends only on this value (Δ_3).This means that The difference (XOR) between the odd bits of any two states depends directly on the difference in last 3 bits of $\{n_R\}$, this weakness is exploited in the card-only attack.

E. Weakness 5: Use of Parity Bits Encrypted by Repeated Keystream

It is a weakness in the communication protocol, where according to the ISO-14443-A standard, every byte sent is followed by a parity bit for error detection, anyway, the Mifare Classic computes the parity bit over the plaintext, instead of the ciphertext and both the parity bit and the first bit of the next plaintext byte are encrypted with the same keystream bit as shown in figure 4. The fact that the keystream bits are repeatedly used implies that the ciphertext reveals linear relations (one relation per pair of ciphertext bits corresponding to the parity bit of the current byte and the first bit of the next byte). This weakness is exploited in the nested authentication attacks to reduce the uncertainty of new tag nonces used for authenticating access to multiple sectors [3, 4].

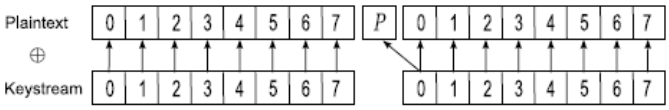

Figure 4: Encryption of parity bit in Mifare Classic protocol [1]

F. Weakness 6: Information Leak from error code

Also, another weakness in the communication protocol, that is during the authentication Protocol, when the reader sends $\{n_R\}$, and $\{a_R\}$, the tag first, checks the parity bits before checking a_R. If one of the eight parity bits is incorrect, the tag will not respond. However, if all eight parity bits are correct, but the response a_R is incorrect, the tag will respond with a 4-bit encrypted error code {0x5} to indicate a transmission error. This happens even the reader has not been authenticated and hence cannot be assumed to be able to decrypt the encrypted error message. If we combine the error code 0x5 with its encrypted version {0x5} by using the bitwise exclusive-or operation, we can recover four keystream bits. This weakness is exploited in the card-only attack.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 6, Issue 10, October 2016)

G. Weakness 7: Card Identification Information Leak from Tag in the Anti-Collision Phase

In the anti-collision stage of the communication protocol, the reader is allowed to select a particular tag to communicate with. It prevents communication collisions by ensuring that different tags do not respond to the reader at the same time. It begins when reader sends the "Anti-collision command" subsequently, the tag broadcasts its 32-bit unique identification number (u_{id}).The anti-collision stage is not cryptographically protected. Therefore, an attacker with an ISO 14443-A reader can possibly get a tag identification number (u_{id}) [7].

Generally, the unique identification number of the radio frequency identification (RFID) tags cannot be considered a security contribution factor [8], but in this paper we consider it as a weakness for the sake of completeness.

H. Weakness 8: 48-Bit Secret Key of CRYPTO-1 is Short

The brute-force attacks can be considered less threatening because it takes years of computation on an ordinary PC. For example, an online brute force attack on the key can be done, with 5ms time period for each attempt because of the communication delay, so an exhaustive key search would then take (2^{48}x 5ms), which equals about 44,627 years. On the other hand, when the cryptographic algorithm is known, an off-line brute force attack can be mounted using a few recorded sessions of authentication, so with expensive supercomputers or dedicated hardware of high cost this would take reasonably short amount of time [5], accordingly, the computational power required by brute-force attacks is way beyond the reach of an ordinary person [9].

I. Weakness 9: Encrypted Tag Nonce of Multiple Sectors Authentication Protocols

When a reader is already authenticated to certain sector in a tag, the protocol of a subsequent authentication for a new sector differs slightly from the first one in that the tag nonce will be encrypted by a key(ks_0) driven from the new sector key before transmitted to the reader. Since the first tag nonce was sent without encryption, and the timing between two authentication attempts can be known, the attacker can guess the second tag nonce and execute bitwise exclusive-or operation with the encrypted second tag nonce to recover the 32 bits of keystream with high accuracy [10].

III. Attacks On MIFARE Classic

In recent years many weaknesses of the Mifare Classic have been exploited for different types of attacks, we divide them into six categories, which include key recovery attack using genuine intercepted authentication session [1,2], card emulation attack [1], card only attack [1,2,9], attacks on genuine session and genuine Tag [2,5], attacks on genuine reader [2,3], and finally multiple sector attacks [2,3,4,9].

A. Attack 1: Key Recovery Using Genuine Intercepted Authentication Session

This attack is aimed to recover the sector key used to encrypt intercepted authentication between genuine tag and genuine reader. The attack consists of two phases where, the offline phase involves the key recovery after the communication has been intercepted in the online phase. The following steps describe this attack in greater details:

1. Eavesdrop the communication between genuine tag and genuine reader and record the authentication session
2. From the recorded authentication session, we are able to identify u_{id}, n_T, $\{a_R\}$,and $\{a_T\}$.
3. According to the Known value of n_T, calculate a_R, and a_T by using equations (1), (2).
4. Get ks_2 by doing the bitwise exclusive-or operation for a_R and $\{a_R\}$, similarly get ks_3 from $a_T \oplus \{a_T\}$.
5. By using Divide-and-Conquer technique ,which is explained in weakness 2, the attack first recover the 48-bit LFSR internal state of CRYPTO1 at a time immediately after all the bits of n_T and n_R have been fed into the LFSR, where this is the state from which the first bit of ks_2 is produced.
6. By exploiting weakness 3, the secret key is obtained by the LFSR rollback to the initial state which is the secret key.

The impact of this attack is that all data from the sector can be obtained with secret key of this sector.

B. Attack 2: Card Emulation

The objective of this attack is to emulate a tag and fool the genuine reader into thinking that it is communicating with the actual tag, to perform the card emulation, we first need to obtain a complete communication that normally takes place between the genuine tag and genuine reader from the first attack, then program our tool to behave like the card. The steps are concluded as follows:

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 6, Issue 10, October 2016)

1. Perform attack 1 and get the secret key of specific sector and the identification u_{id} of the genuine card to be emulated.
2. By using the secret key of that specific sector in the genuine card, we are able to read the corresponding data in this sector.
3. Use the card identification u_{id} to program our tool to handle the anti-collision phase using u_{id} of the card we are emulating.
4. Use the sector key, card identification u_{id} , tag nonce n_T that will be sent by the tool, and reader nonce n_R that will be decrypted to program our tool to perform the authentication requests from the reader by deciding what to respond (encrypted a_T) taking into consideration that $a_T = \text{suc}^{96}(n_T)$.
5. Use the sector key, card identification u_{id} ,tag nonce n_T , and decrypted reader nonce n_R to encrypt and send the data obtained in step 2, at this point ,we have emulated the card partially (for only one sector).
6. Repeat steps from 1 to 5, for all sectors if we wish to emulate completely the genuine card.

C. *Attack 3: Key Recovery Using Card Only*

The most practical and safe scenario is the card only attack because it needs only access to genuine card by using fake reader without exposure the attacker to the authorities as, most gates or doors protected by additional human or camera surveillance that discourages the attacker from attempting the attack. The most effective known attack in the tag-only scenario is the differential attack, which requires about 10 seconds of average on-line time in order to reconstruct the secret key from the card [2]. The steps of the differential attack are as follows :

1. Starting with the on-line stage, We attempt authentications to the tag with a random n_T , random 8-byte ($\{n_R\}$, $\{a_R\}$) ciphertext and 8 random encrypted parity bits to get 3 error message where, on average, 256 authentications are required to get one successful 4-bit error message.
2. Identify and group each of n_T, $\{n_R\}$, $\{a_R\}$, and the 8 random encrypted parity bits that leads to the successful error messages and select the most frequent tag nonce n_T group.
3. We repeatedly attempt authentications with the tag but only continue if the tag nonce is the same as our chosen n_T . We will then keep the first 29 bits of $\{n_R\}$ constant, the entire $\{a_R\}$constant, and the first three encrypted parity bits constant.

   We will try in order each of the seven possible nonzero 3-bit modification for $\{n_R\}$ and assign random values for the last five encrypted parity bits. We should get a 4-bit error message for each possible combination for $\{n_R\}$.
4. From now, the attack is offline. We first recover the keystream used to encrypt the transmission error message, by simply taking the exclusive-or of the encrypted error message with the known plaintext 0x5.
5. By exploiting Weakness 2, we Start with a search space of 2^{21} and remove those that do not generate even bits $ks_3(0)_1$ and $ks_3(2)_1$ (first and third bit of ks_3 from the first message). This should reduce the search space by four.
6. By exploiting weakness 4, we perform XOR the remaining elements from last step (2^{19}) with the state difference that corresponds to the difference in $\{n_{R1}\}$ and $\{n_{R2}\}$ to get the possible next states.
7. We next remove those possible next states that do not generate $ks_3(0)_2$ and $ks_3(2)_2$, which reduces the search space by four, so we have 2^{17} possibilities of 21bits.
8. We repeat step 6 and 7 for the remaining six error messages. We should finally get approximately 2^5 possible states each of 21 bits.
9. We repeat the steps (5-8) above for the odd bits $ks_3(1)_1$ and $ks_3(3)_1$, $ks_3(1)_2$ and $ks_3(3)_2$, and so on to get the other 2^5 possible states.
10. Combining the two lists of 2^5 possible states, to get a list of 2^{10} states of 42-bits wide, then We expand the list to 2^{16} states of 48-bits wide.
11. By exploiting Weakness 3, we rollback each of the 2^{16} possible states to obtain the corresponding 2^{16} possible keys.
12. For each of the 2^{16} possible keys check if it generates all eight sets of parity bits, stopping immediately once we find the key that does verify this condition.

D. *Attack 4: Attacks on Both Genuine Session and Genuine Tag*

This attack demands access to a genuine authentication session and then access to a genuine tag, by a fake reader. It neither requires knowledge of the encryption algorithm nor aims at reconstructing the secret key. It aims at reconstructing portions of the keystream, which are repeatedly used for encrypting data in fake sessions initiated by a fake reader, so we need complete control over the reader (Proxmark III) and access to a genuine card. The attack consists of the following steps:

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 6, Issue 10, October 2016)

1. Eavesdrop and record the communication between a genuine reader and a genuine card.
2. Perform authentication between fake reader and genuine card, and make sure that the card will use the same keystream as in the recorded communication, which can be achieved by using the same recorded session n_T, and $\{n_R\}$, however the same tag nonce n_T can be obtained by using either the periodic query or the reset query techniques, due to weakness 1 of the tag pseudorandom number generator, and the same encrypted reader nonce $\{n_R\}$is granted by replaying the second pass of the authentication from the recorded communication.
3. By changing the block number in a read command that will enable the card to receive a command for which we know plaintext of the encrypted response such as tag identification and manufacturer data.
4. For each segment of known plaintext, compute the corresponding keystream segment by using bitwise XOR operation between known plaintext and received ciphertext.
5. Use this keystream to partially decrypt the trace obtained in step1.
6. Repeat the procedures for any encrypted data which is received by the reader.

It is worth mentioning that the general idea of this attack can be concluded as follows:

The plaintext P_1 in the communication is bitwise XOR-ed with a keystream K which gives the encrypted data C_1.When it is possible to use the same keystream on a different plaintext P_2 to get C_2 and either P_1 or P_2 is known, then both P_1 and P_2 are revealed.

E. Attack 5: Key Recovery Using Genuine Reader

This attack requires access to a genuine reader by a fake tag. In this case, the authentication session cannot be completed, because the fake tag cannot be successfully authenticated to the reader as it does not know the key and cannot send the right encrypted tag response $\{a_T\}$.The attack can be executed according the following steps:

1. Use a fake tag to communicate with a genuine reader.
2. During the first pass of the authentication between the reader and the tag, a known tag nonce n_T is sent by the fake tag, use this value to calculate a_R according to equation 1 where: $a_R = \text{suc}^{64}(n_T)$
3. The fake tag will receive $\{n_R\}$, and $\{a_R\}$, which are sent by the genuine reader.
4. Calculate ks_2 using bitwise XOR operation of $\{a_R\}$ (obtained from step 3) and a_R (calculated at step 2).
5. We may need to repeat steps (2-4) in order to collect more data from different partial authentication sessions.
6. Starting with the 32 bit key stream of ks_2, use Divide-and-Conquer technique, which is explained in weakness 2, the attack first recover the 48-bit LFSR internal state of CRYPTO1 at a time immediately after all the bits of n_T and n_R have been fed into the LFSR, where this is the state from which the first bit of ks_2 is produced.
7. By exploiting Weakness 3, make rollback to get the initial state which is the secret key.

F. Attack 6: Multiple Sector Attack

The attacks which are discussed so far, relate to authentication for only one sector. When a reader is already having authenticated to the first sector of the tag, the protocol of a subsequent authentication for a new sector differs slightly from the initial one in the following two manners:

1. The new authentication block is encrypted with the previous sector key.
2. The tag nonce will be encrypted $\{n_T\}$ by the preceding keystream segment (ks_0) generated from the new secret key before transmitted to the reader.

By assuming that the key for the first sector has been recovered by one of the scenarios described in this section which include using genuine intercepted authentication session (attack 1), using card only (attack 3), or using genuine reader (attack 5), the objective is then to reconstruct the secret keys for other sectors so it is required to gather data from an authentication session for multiple sectors, whereas this is possible only in attack1, and attack 3, but in attack 5 the fake tag cannot encrypt a new n_T with the correct key for the next sector and hence attack 5 is not successful.

This attack is known as nested authentication attack and depend on guessing the correct value for the new n_T, which has only 2^{16} values. An important factor which further reduces the uncertainty is that the first three parity bits of the tag nonce n_T, denoted by p_0, p_1, and p_2, are encrypted with the keystream bits that are also used to encrypt bits n_{T8}, n_{T16}, and n_{T24} of n_T respectively, so from the communication we can observe if $\{p_0\}$ and$\{n_{T8}\}$ are equal or not, then we can preserve the same relation between p_0, and n_{T8}, and similarly for the other two parity bits (p_1, p_2), where each relation reduces the plaintext uncertainty by 1 bit therefore, the initial uncertainty of n_T given $\{n_T\}$ is (16-3) bits, another factor is the known timing between two authentication attempts.

We can present steps for multiple sector attack in attack 1 scenario (key recovery using genuine session) as follows:

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 6, Issue 10, October 2016)

1. Since, the first sector key was reconstructed; it is used for decrypting the next authentication command, to know what the next sector is.
2. From the second authentication session, get $\{n_T\}$, $\{a_R\}$, and $\{a_T\}$,since we have ten parity bits information for $\{n_T\}$,$\{a_R\}$, and $\{a_T\}$ it is clear that the uncertainty of n_T, is only 6 bits, so that the attacker needs to check less than 2^6 values of n_T.
3. For any guessed value of n_T calculate the corresponding values of a_R and a_T using equations (1) and (2), then the 96-bit keystream segment (ks_0, ks_2, ks_3) is obtained by using bitwise XOR operation according to equations (4),(5), and (6) as follows:

$$ks_0 = n_T \oplus \{n_T\} \quad (4)$$
$$ks_2 = a_R \oplus \{a_R\} \quad (5)$$
$$ks_3 = a_T \oplus \{a_T\} \quad (6)$$

4. The key for the next sector can be reconstructed from the corresponding 96-bit keystream segment (ks_0, ks_2, ks_3) by the techniques described in section II (Divide-and-Conquer then rollback techniques), provided that the guess is correct.
5. The keys of other sectors can be reconstructed in a similar fashion from already recovered keys of the previous sectors.

Now, the steps for multiple sector attack in attack 3 (key recovery using card only) scenario is presented as follows:

1. After successfully reconstructed the secret key for the first sector by the fake reader, then proceeds with the authentication for the next sector by sending the authentication command encrypted with the key for the first sector.
2. The fake reader will receive the encrypted tag nonce $\{n_T\}$ for the new sector.
3. It is clear that the uncertainty of n_T, is only 13 bits, so that the attacker needs to check less than 2^{13} values of n_T, which may be significantly reduced by using the measured timing between the two authentication sessions.
4. For each guessed value of n_T, a corresponding 32-bit keystream segment (ks_0) is obtained by using bitwise XOR operation between n_T, and $\{n_T\}$.
5. Each 32-bit keystream segment leads to about 2^{16} candidate keys for the next sector that can be reconstructed by the techniques described in section II.
6. By using at most two additional fake authentication sessions for multiple sectors initiated by the fake reader, the correct key for the next sector is found by computing the intersections between the obtained sets of candidate keys.

An important observation is that the keys of other sectors in attack 3 scenario can be reconstructed much easier than for the first sector.

The previous attacks could be performed without the need to very specialist or hard to obtain equipment or unavailable software [11], for this reason, Mifare Classic should be pushed to improved security solutions.

IV. Proposed Solutions For Improving The Weaknesses Of MIFARE Classic

In this section, we propose many solutions for improving many weaknesses and consequently, block the known attacks. It can be classified into three main categories as follows:

1. *Solution 1:* Improving weakness 1 of pseudo-random number generator, and Weakness 9 of the communication protocol.
2. *Solution 2:* Improving weakness 2, and weakness 3 of cryptographic cipher.
3. *Solution 3:* Improving weakness 5, and weakness 6 in the Mifare Classic communication protocol.

A. Solution 1: Pseudo-Random Number Generator

The 32-bit tag nonce is actually generated by a 16-bit LFSR that runs in a deterministic cycle takes about 618 ms, which is short time, in addition its state starts from the same known one after powering up the tag ,besides it is sent to reader without encryption for the first sector.

All these issues have been exploited in attack 1, attack 4, attack 5, and attack 6. To enhance these weaknesses, we propose three modifications as follows:

1. Returning back to figure 3 and equation (3), we notice that the feedback function depends on 4 bits of the second half of the tag nonce (x_{16}, x_{18}, x_{19}, x_{21}) that causes entropy of (2^{16}-1). If the feedback function will be modified to become dependable on both halves of the tag nonce as shown in equation (7) that will increase the entropy to (2^{32} -1), consequently repeating same state will takes about 11.26 hours which is significant obstacle in using periodic query in attack 4, also this increase the online time horribly in attack 3 when periodic query is used to obtain fixed tag nonce.

$$L(x_{16}\, x_{17} \ldots x_{31}) = x_0 \oplus x_1 \oplus x_{27} \oplus x_{28} \quad (7)$$

Also this leads to increase the uncertainty in attack 6 to become 2^{32} instead of 2^{16}. In other words it will behave as 32-bit LFSR.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 6, Issue 10, October 2016)

2. An important modification to increase the randomness is to enforce the tag nonce to start from random state each time the tag will power up, that will thwart using of reset query technique to obtain fixed tag nonce in attack 3, and in attack 4.
3. Send the tag nonce n_T encrypted by 32-bit keystream segment (ks_0) from the first sector, this modification will definitely stop attack 1, and attack 5 because in attack 1 the plaintext of the encrypted tag nonce cannot be obtained but the uncertainty in getting n_T will drop by a value of 3 due to the first three parity bits associated with the $\{n_T\}$, while in attack 5 the fake tag cannot encrypt n_T with the correct key.

   Also attack 6 will become less dependable on measured timing between two authentication sessions, unless the key for the first sector is used to decrypt $\{n_T\}$ corresponds to this sector.

This modification is related to the communication protocol, but it is discussed here as it concerns the tag nonce.

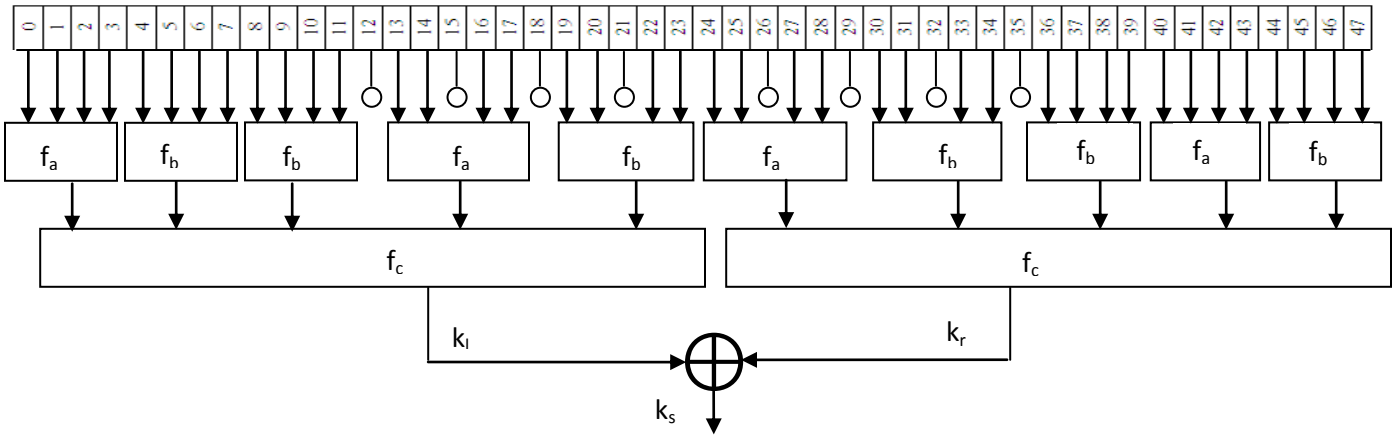

Figure 5: Keystream generation after modification the inputs to be 40 input bits applied to 2 identical nonlinear filter generators.

The power of the three preceding modifications is to apply them together; this will stop attack 1, attack 4, and attack 5. Attack 3 will be applicable with only random tag nonce. Also attack 6 will need massive amount of collecting data to deal with the uncertainly of 32 bits instead of 16 bits.

B. Solution 2: Cryptographic Cipher

As we discussed before in section 2, there are two weaknesses related to the cryptographic cipher, known as weakness 2 ,and weakness 3, where weakness 2 resulted because only odd state bits used as inputs to the filter generator to generate keystream and this allows the attacker to obtain a specific LFSR state after knowledge of some consecutive keystream bits, while weakness 3 resulted because the first nine leftmost bits of LFSR are not used as inputs to the filter generator, this permits the attacker to obtain the initial LFSR state which is the secret key after the knowledge of a particular state.

To enhance both weaknesses, we propose a modification which is increasing the number of inputs to the nonlinear filter generator to become 40 inputs, that are applied to two identical nonlinear functions, where each function is the same as the original nonlinear filter generator, then the two outputs from the two identical functions are combined together using bitwise XOR operation. According to figure 5, there are 40 output bits that consists of 20 odd bits ,and 20 even bits from the 48-bit LFSR which represent all the outputs except 8 bits (bits number 12,15,18,21,26,29,32,and 35). These 40 bits are applied to two identical nonlinear filter generator resulting outputs k_l and k_r . These two outputs are combined together using XOR operation to finally result keystream bit k_s as clarified in equation 8.

$$k_s = k_l \oplus k_r \qquad (8)$$

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 6, Issue 10, October 2016)

Undoubtedly, this modification will stop attack 1, attack 3, attack 5, and attack 6 because a Divide-and-Conquer technique to get specific LFSR state and rollback technique to obtain the secret key cannot be applied.

C. Solution 3: Communication Protocol

There are two weaknesses related to the communication protocol which are weakness 5, and weakness 6.

Firstly, according to weakness 5, the protocol computes the parity bit over the plaintext and both the parity bit and the first bit of the next plaintext byte are encrypted with the same keystream bit as discussed before in section II.

Our modification is to calculate parity bits over the plaintext and use either k_l or k_r computed from the two identical nonlinear generators illustrated in figure 5 to encrypt each parity bit. The first bit of the next plaintext byte is encrypted with the key k_s which is dependable on both of the two nonlinear function outputs k_l and k_r. Of course this modification will enhance the current issue which is to observe the relation between the encrypted parity bit and the encrypted bit of the next byte and to keep the same relation between the corresponding plaintext parity bit and first bit of the next byte. This weakness is exploited in attack 6 to reduce the uncertainty of new tag nonces used for authenticating access to multiple sectors, as a result, our modification will make attack 6 less applicable as it will depend only on the known timing between two authentication attempts.

Secondly, Concerning weakness 6 which happens during the three-pass mutual authentication protocol ,where if all the eight parity bits of $\{n_R\}$,$\{a_R\}$ are correct, but the response $\{a_R\}$ is incorrect, the tag will respond with an encrypted 4-bit error code, even though the reader itself has not been authenticated and cannot be assumed to be able to decrypt the message. This error message can be combined with the known plaintext 0x5 by using bitwise XOR operation, and subsequently revealing 4 keystream bits.

Our solution can be one of the two following choices:

1. When all the parity bits of $\{n_R\}$,$\{a_R\}$ are correct, but the response $\{a_R\}$ is incorrect ,the tag does not respond at all like the situation that the parity bits are incorrect.
2. The tag sends clear plaintext error message (0x5) in both following cases:
    a. Parity bits are incorrect.
    b. $\{a_R\}$ is incorrect.

It is very clear that this modification will stop attack 3 because no keystream bits will be revealed by using encrypted error code.

V. Conclusions

In this paper, the Mifare Classic history presents further strong evidence that use of proprietary encryption algorithm and security by obscurity principle are design concepts that should be avoided in cryptographic systems because, it covers up weaknesses rather than gives an additional layer of protection, in addition reverse engineering techniques have successfully discovered the weaknesses which are exploited through different known attacks that are discussed previously. Our contribution is to improve the security level by find simple solutions to many weaknesses and hence block all the known attacks when we apply all the proposed solutions together.

Actually, we have proposed six modifications classified in three main categories which are solution 1, solution 2, and solution 3. These solutions are capable of halting all the known attacks discussed in section III taking into consideration that attack 2 will be infeasible if attack 1, and attack 3 is thwarted because card cannot be emulated unless card ID ,and secret key are revealed. Table I clarifies the proposed solutions with its effect on the known attacks, accordingly we can halt specific attack by applying one of the corresponding solutions that proposed to completely stop (third column), except attack 2 to ensure totally blocking, we should apply solution 2 or apply solution 1 and solution 3 together. Any way it is recommended to apply all the three solutions together for a comprehensive guarantee to stop all known attacks. Moreover, we recommend that second degree of security is used to encrypt all the data before storing it at the card except the card identification as the attacker can get the clear form of the identification number and combined it with the encrypted form to obtain the used key.

Table I
The effect of the proposed solutions on the different known attacks

	Attack	Proposal Solutions to be Less Applicable	Proposal Solutions to Completely Stop
1	Attack 1	–	Solution 1, or Solution 2
2	Attack 3	Solution 1	Solution 2,or Solution 3
3	Attack 4	–	Solution 1
4	Attack 5	–	Solution 1 ,or Solution2
5	Attack 6	Solution 1 , or Solution 3	Solution 2
6	Attack 2	–	Solution 2, or Solution 1, and Solution 3

Acknowledgment

First, thanks god for finishing and completing this paper, we would like to say Elhamdlillah. We want to thank Ahmed Hossam for what he did. We are really grateful to him.

REFERENCES

[1] Wee Hon Tan, "Practical Attacks on the MIFARE Classic" Submitted in partial fulfillment of the requirements for the MSc. Degree in Computing Science of Imperial College London; September 2009.

[2] Jovan Dj. Goli´, "Cryptanalytic Attacks on MIFARE Classic Protocol", E. Dawson (Ed.): RSA 2013, LNCS 7779, pp. 239–258, 2013, Springer-Verlag Berlin Heidelberg 2013.

[3] F.D. Garcia, G. de Koning Gans, , R. Muijrers, , P. van Rossum, R. Verdult, R.W. Schreur, and B. Jacobs, "Dismantling MIFARE Classic", In S. Jajodia, J. Lopez, (Eds.) ESORICS 2008, LNCS vol. 5283, pp. 97–114. 2008.

[4] F.D. Garcia, P. van Rossum, R. Verdult, and R.W. Schreur, "Wirelessly Pickpocketing a Mifare Classic card", In Proceedings of the 2009 30th IEEE Symposium on Security and Privacy, S&P 2009, pp. 3–15. IEEE Computer Society, Washington, DC (2009).

[5] G. de Koning Gans, J.-H. Hoepman, and F.D. Garcia, "A Practical Attack on the MIFARE Classic", In G. Grimaud, and F.-X. Standaert, (Eds.) CARDIS 2008. LNCS vol. 5189, pp. 267–282, 2008.

[6] Heiko Knospe and Kerstin Lemke-Rust, "Towards Secure and Privacy-Enhanced RFID Systems", RFID Systems: Research Trends and Challenges, 2010 John Wiley and Sons Ltd. ISBN: 978-0-470-74602-8.

[7] Ton van Deursen, "50 Ways to Break RFID Privacy", S. Fischer-Hubner et al. (Eds.): Privacy and Identity 2010, IFIP AICT 352, pp. 192–205, 2011. IFIP International Federation for Information Processing 2011.

[8] Gursel Duzenli, "A new security approach for public transport application against tag cloning with neural network-based pattern recognition", Published online: 7 February 2015, Neural Comput and Applic (2015) 26:1681–1691 DOI 10.1007/ s00521-015-1837-8.

[9] Yi-Hao Chiu, Wei-Chih Hong, Li-Ping Chou, Jintai Ding, Bo-Yin Yang, and Chen-Mou Cheng, "A Practical Attack on Patched MIFARE Classic", D. Lin et al. (Eds.): Inscrypt, LNCS 8567, pp. 150–164, 2014. DOI: 10.1007/978-3-319-12087-4-10.

[10] Flavio D. Garcia, and Bart Jacobs, "The Fall of a Tiny Star", P.Y.A. Ryan et al. (Eds.): Kahn Festschrift, LNCS 9100, pp. 69–87, 2016. DOI: 10.1007/978-3-662-49301-4-5.

[11] Keith E. Mayes, and Carlos Cid, "The MIFARE Classic story", Information Security Technical Report I5 (2010) 8-I2, 2010 Elsevier Ltd. DOI:10.1016/j.istr.2010.10.009.