KRYPTOANALIZA

Seminarium: Bezpieczeństwo komputerowe

ODZYSKIWANIE KLUCZA

- Schemat szyfrowania symetrycznego jest odporny na odzyskiwanie klucza w ramach ataku z wybranym plaintextem (secure against key recovery under chosen plaintext attack) jeśli dla każdego nonce-respecting PPT algorytmu A zachodzi

$$P[A^{Enc(K\ldots)} \rightarrow K] = negl(s)$$

SŁOWNICZEK

- PPT algorithm = probabilistic polynomial-time algorithm

SŁOWNICZEK

- PPT algorithm = probabilistic polynomial-time algorithm
- Nonce - liczba której możemy użyć tylko raz

SŁOWNICZEK

- PPT algorithm = probabilistic polynomial-time algorithm
- Nonce - liczba której możemy użyć tylko raz
- nonce-respecting - algorytm którego używamy nie może użyć nonca dwa razy przy zapytaniach szyfrujących ale może przy zapytaniach deszyfrujących

SŁOWNICZEK

- PPT algorithm = probabilistic polynomial-time algorithm
- Nonce - liczba której możemy użyć tylko raz
- nonce-respecting - algorytm którego używamy nie może użyć nonca dwa razy przy zapytaniach szyfrujących ale może przy zapytaniach deszyfrujących
- negl(s) - funkcja f jest negligible (mało istotna) jeśli przy s (parametr bezpieczeństwa) dążącym do nieskończoności dla każdego n jest $O(s^{-n})$

ODZYSKIWANIE KLUCZA

- Schemat szyfrowania symetrycznego jest odporny na odzyskiwanie klucza w ramach ataku z wybranym plaintextem (secure against key recovery under chosen plaintext attack) jeśli dla każdego nonce-respecting PPT algorytmu A zachodzi

$$P[A^{Enc(K\ldots)} \rightarrow K] = negl(s)$$

ODZYSKIWANIE KLUCZA

- Schemat szyfrowania symetrycznego jest odporny na odzyskiwanie klucza w ramach ataku z wybranym plaintextem (secure against key recovery under chosen plaintext attack) jeśli dla każdego nonce-respecting PPT algorytmu A zachodzi

$$P[A^{Enc(K\ldots)} \rightarrow K] = negl(s)$$

- Podobnie gdy możemy wybrać plaintext lub ciphertext

$$P[A^{Enc(K\ldots)Dec(K..)} \rightarrow K] = negl(s)$$

ODZYSKIWANIE KLUCZA

- Złamanie CPA => Złamanie CCA

ODZYSKIWANIE KLUCZA

- Złamanie CPA => Złamanie CCA

- Bezpieczeństwo względem CCA => Bezpieczeństwo względem CPA

ODPORNOŚĆ NA ROZSZYWROWANIE

- Schemat szyfrowania symetrycznego jest odporny na rozszyfrowywanie z wybranym plaintextem (secure against decryption under chosen plaintext attack) jeśli dla każdego nonce-respecting PPT algorytmu A zachodzi

$$P[A^{Enc(K,...)}(N, Enc(K, N, X)) \rightarrow X] = negl(s)$$

Gdzie K jest dowolnym kluczem, N noncem, a X plaintextem

$$P[A^{Enc(K,...)Dec(K,...)}(N, Enc(K, N, X)) \rightarrow X] = negl(s)$$

ODPORNOŚĆ NA ROZSZYFROWYWANIE

- Odzyskanie klucza => rozszyfrowanie

- Bezpieczeństwo względem rozszyfrowywania => bezpieczeństwo względem odzyskiwania klucza

ODPORNOŚĆ NA ROZRÓŻNIANIE

- Schemat szyfrowania symetrycznego jest odporny na rozróżnianie w ramach ataku z wybranym plaintextem (secure under chosen plaintext attack) jeśli dla każdego nonce-respecting PPT algorytmu A zachodzi

$$P[A^{Enc(K...)} \rightarrow 1] - P[A^{\Pi()} \rightarrow 1] = negl(s)$$

- Podobnie gdy możemy wybrać plaintext lub ciphertext

$$P[A^{Enc(K...)Dec(K...)} \rightarrow 1] - P[A^{\Pi(...)\Pi(...)^{-1}} \rightarrow 1] = negl(s)$$

ODPORNOSC NA ROZROZNIANIE

- Bezpieczeństwo względem rozróżniania => bezpieczeństwo względem rozszyfrowywania

RODZAJE ATAKÓW

	Odzyskanie klucza	Rozszyfrowanie	Rozróżnianie
CPA	Najmniej bezpieczne		
CCA			Najbardziej bezpieczne

FUNKCJE HASZUJĄCE

- Wykorzystanie funkcji haszujących:
  - Commitment

$$\begin{aligned} \text{Commit}(X; \text{random}) &= (H(\text{Key}), \text{Key} = X \| \text{random}) \\ \text{Open}(c, X \| \text{random}) &= \begin{cases} X & \text{if } H(X \| \text{random}) = c \\ \perp & \text{otherwise} \end{cases} \end{aligned}$$

  - PNRG

$$\text{Generation} = \text{H}(\text{seed}||\text{counter})$$

  - KDF
  - Przy autentykacji

FUNKCJE HASZUJĄCE

- Wykorzystanie funkcji haszujących:
  - Commitment
  - PNRG
  - KDF
    - Przy autentykacji
- Właściwości gwarantujące bezpieczeństwo:
  - Odporność na kolizje
  - Jednostronność
  - Pseudolosowość

FUNKCJE HASZUJĄCE

- Collision attack

  znajdujemy takie x i y, że $x \neq y$ i $h(x) = h(y)$

FUNKCJE HASZUJĄCE

- Collision attack

  znajdujemy takie x i y, że $x \neq y$ i $h(x) = h(y)$

- First preimage attack

  mając x znajdujemy takie y, że $x = h(y)$

FUNKCJE HASZUJĄCE

- Collision attack

  znajdujemy takie x i y, że $x \neq y$ i $h(x) = h(y)$

- First preimage attack

  mając x znajdujemy takie y, że $x = h(y)$

- Second preimage attack

  mając x znajdujemy takie y, że $x \neq y$ i $h(x) = h(y)$

PARADOKS DNIA URODZIN

- Jeśli będziemy brać niezależnie losowo (z prawd. jednostajnym) liczby z {1,..,N} po $\theta\sqrt{N}$ razach prawdopodobieństwo, że choć jedna z liczb się powtórzy to:

$$1 - \frac{N!}{N^{\theta\sqrt{N}}(N-\theta\sqrt{N})!} \to_{N\to\infty} 1 - e^{-\frac{\theta^2}{2}}$$

PARADOKS DNIA URODZIN

- Jeśli będziemy brać niezależnie losowo (z prawd. jednostajnym) liczby z {1,..,N} po $\theta\sqrt{N}$ razach prawdopodobieństwo, że choć jedna z liczb się powtórzy to:

$$1 - \frac{N!}{N^{\theta\sqrt{N}}(N-\theta\sqrt{N})!} \to_{N\to\infty} 1 - e^{-\frac{\theta^2}{2}}$$

- Dla N=365 mamy:

$\theta\sqrt{N}$	10	15	20	25	30	35	40
θ	0.52	0.79	1.05	1.31	1.57	1.83	2.09
probability	12%	25%	41%	57%	71%	81%	89%

ATAK A DŁUGOŚĆ KLUCZA

primitive	attack	complexity	parameter n
encryption	key recovery	$\Theta(2^n)$	key length
hash function	preimage attack	$\Theta(2^n)$	hash length
	collision	$\Theta(2^{\frac{n}{2}})$	hash length
MAC	key recovery	$\Theta(2^n)$	key length

SZYFR BLOKOWY

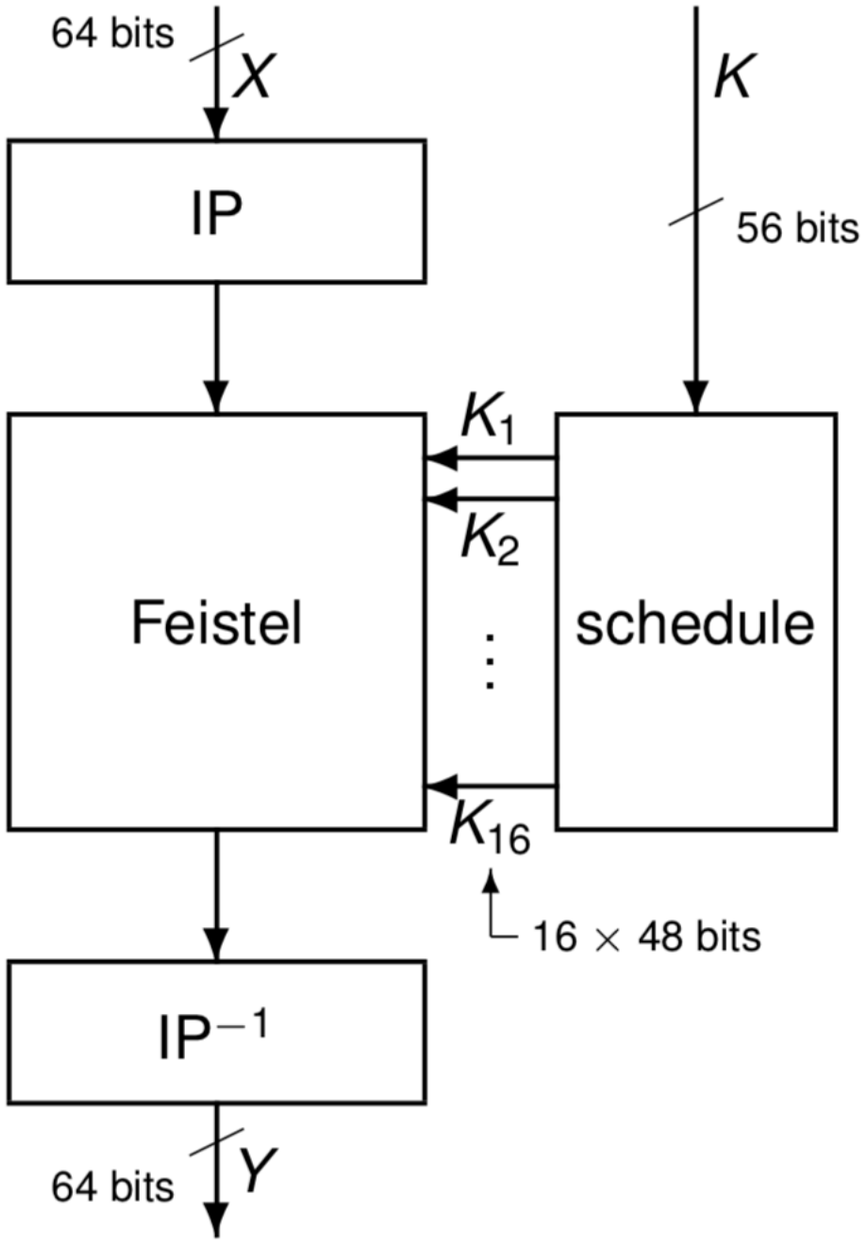
- Szyfr blokowy to taka krotka ($\{0,1\}^k$, D, Enc, Dec) gdzie $\{0,1\}^k$ jest kluczem, D tekstem do zaszyfrowania, a Enc i Dec to dwa (wielomianowe względem parametru s) deterministyczne algorytmy, spełniająca

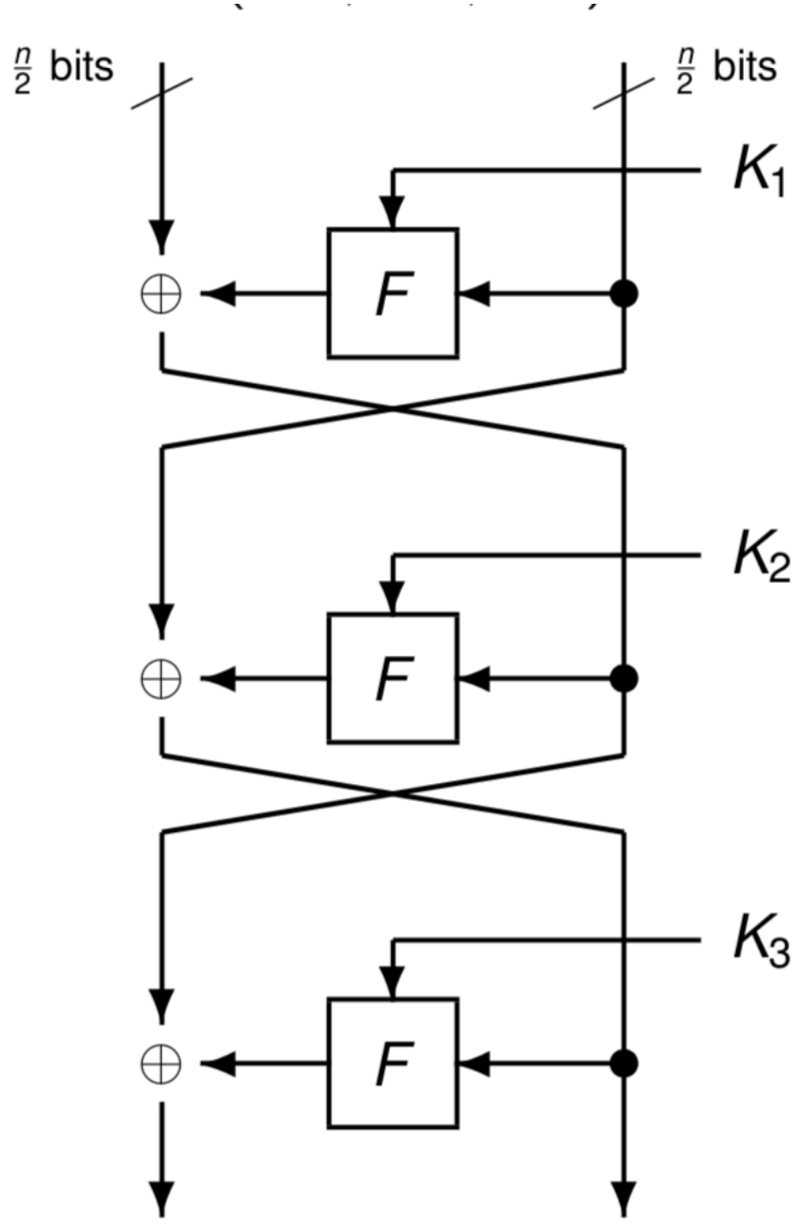
$$\forall s \forall K \in \{0,1\}^k \ \forall X \in D \ Dec(K, Enc(K, X)) = x$$

DES – DATA ENCRYPTION STANDARD

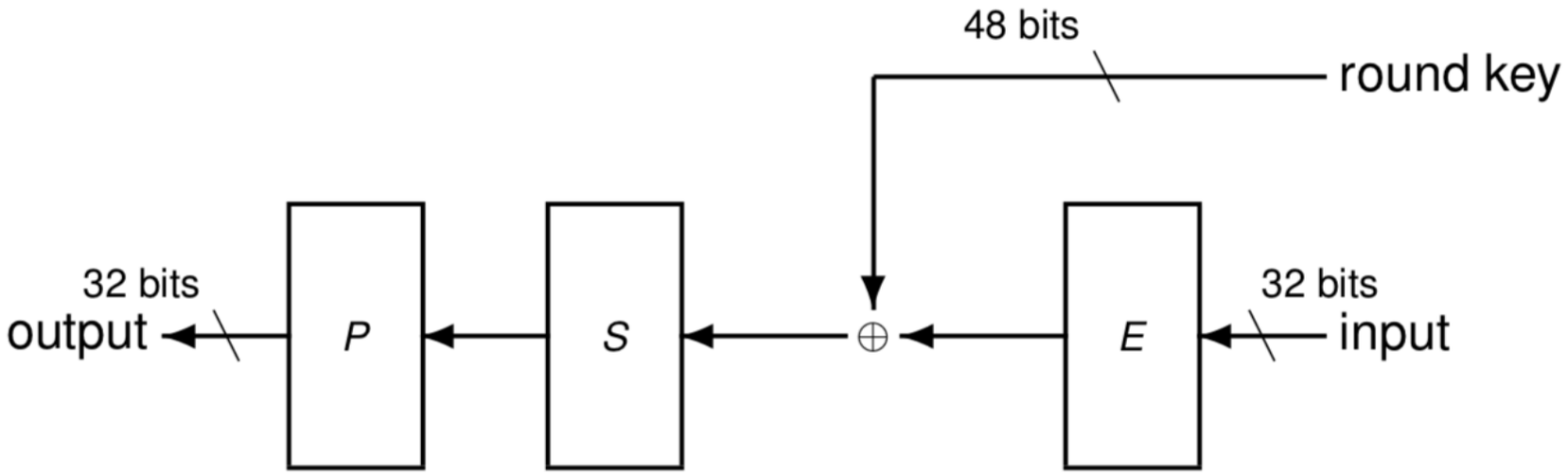
- Powstały w IBMie w latach ’70 (oparty na LUCIFER stworzonym przez Horsta Feistela)
- Złamany w 1997
- Szyfr blokowy z 64-bitowymi blokami
- Klucz z 56 efektywnymi bitami

DES


64 bits
X
IP
K
56 bits
K1
K2
Feistel
schedule
K16
16 × 48 bits
IP−1
64 bits
Y


n/2 bits
n/2 bits
K1
F
K2
F
K3
F

DES



- E - ekspansja 32 do 48 bitów (za pomocą permutacji)
- S - osiem 6 na 4 bitowych S-bloków (substitution boxes)
- P - permutacja

AES

- 128-bitowy blok 4x4 Bajty
- Zależnie od długości klucza (128b, 192b, 256b) mamy 10, 12 albo 14 rund

AES encryption(s, W)
1: **AddRoundKey**(s, W_0)
2: **for** $r = 1$ to $\mathrm{Nr} - 1$ **do**
3: **SubBytes**(s)
4: **ShiftRows**(s)
5: **MixColumns**(s)
6: **AddRoundKey**(s, W_r)
7: **end for**
8: **SubBytes**(s)
9: **ShiftRows**(s)
10: **AddRoundKey**(s, W_{Nr})

ATAK CZASOWY

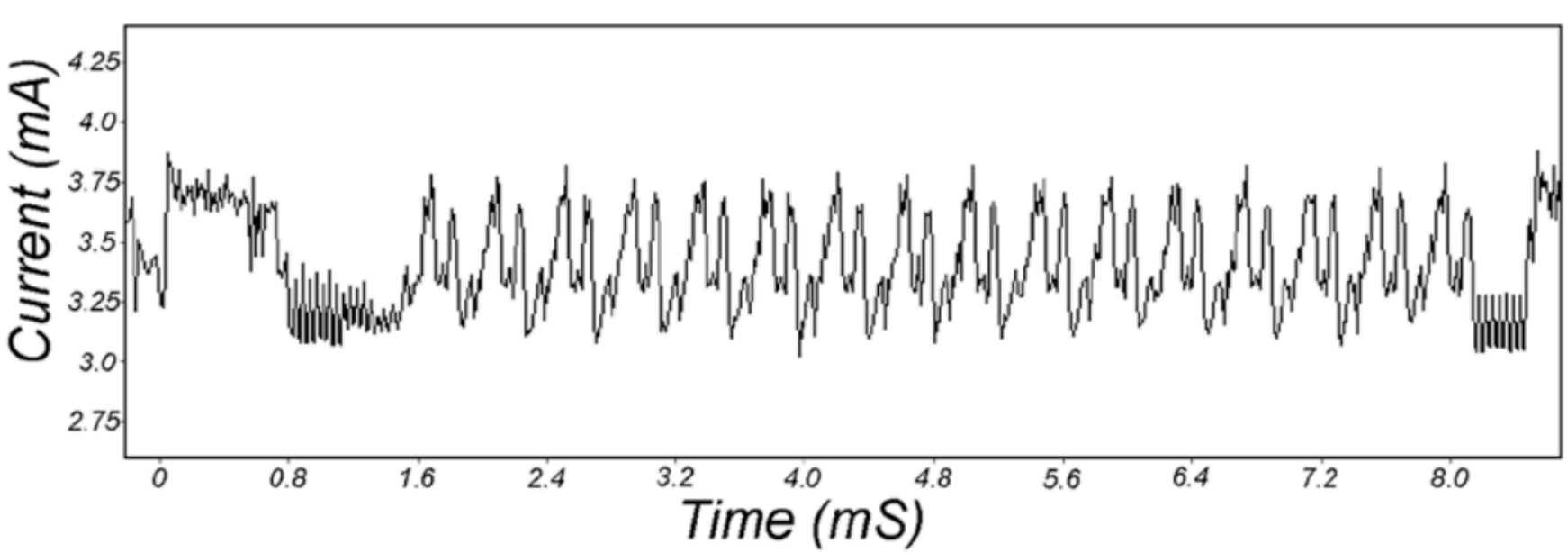
- **Atak czasowy** (ang. timing attack) – atak kryptograficzny, w którym intruz usiłuje skompromitować kryptosystem poprzez analizę czasu, wymaganego do wykonania algorytmów kryptograficznych.

ATAK CZASOWY

- **Atak czasowy** (ang. timing attack) – atak kryptograficzny, w którym intruz usiłuje skompromitować kryptosystem poprzez analizę czasu, wymaganego do wykonania algorytmów kryptograficznych.
- W algorytmie RSA potrzebujemy wyliczać m^d mod n. Jeśli będziemy używać szybkiego potęgowania możemy wyliczyć ile razy musieliśmy mnożyć liczby i dzięki temu uzyskujemy informację o d (klucz prywatny)

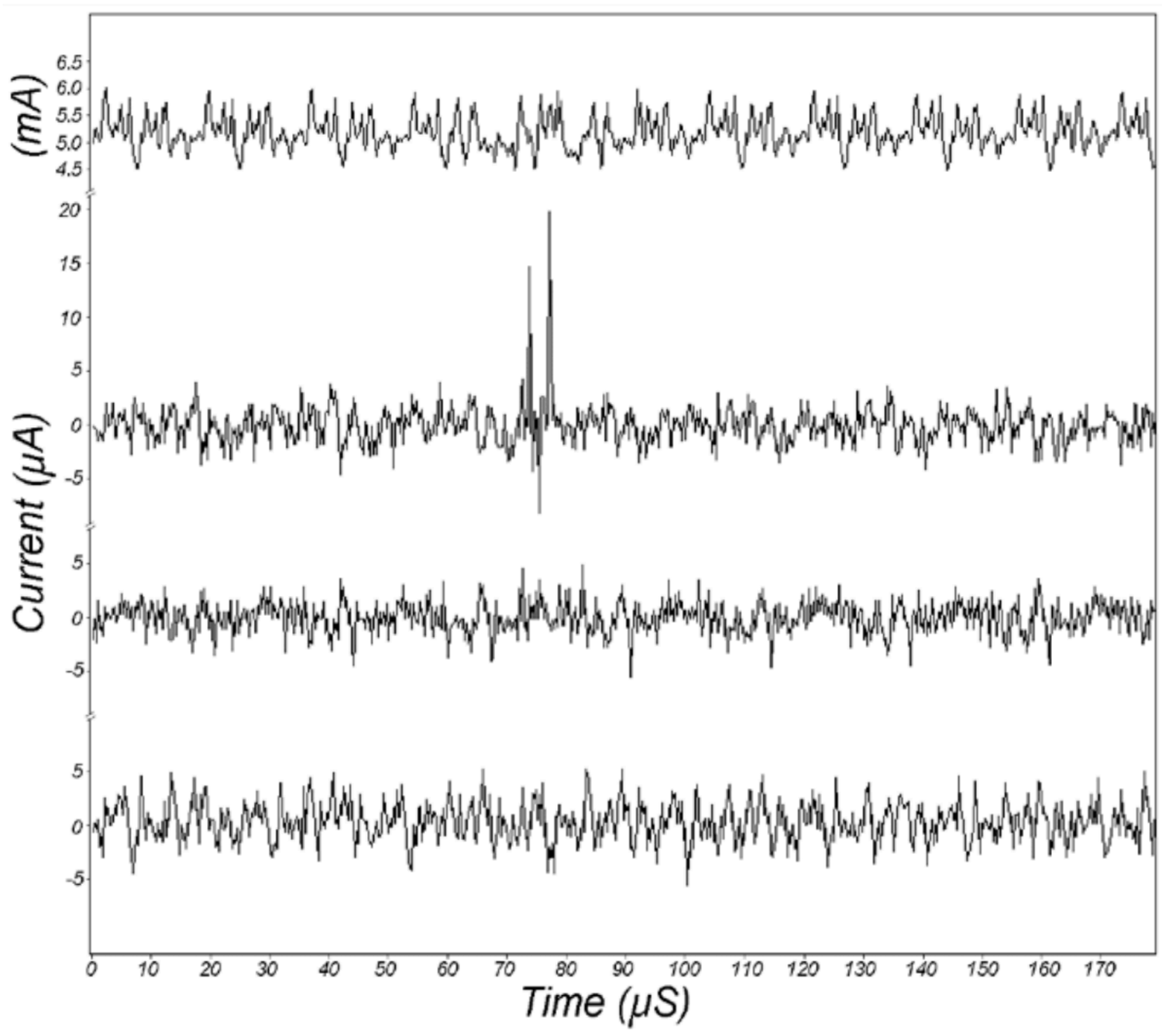
ATAK MOCOWY

- Prosta analiza czasu (DES)



ATAK MOCOWY

- Różnicowa analiza mocy (DES)



HISTORIA

- Kryptoanaliza liniowa - EUROCRYPT ’93
- Kryptoanaliza różnicowa - CRYPTO ’90
- Oba jako przykłady ataku na DES

ATAK LINIOWY

- Atakujący posiada pary (plaintext X, ciphertext Y)
- Próbujemy znaleźć zależność, która zachodzi z dużym prawdopodobieństwem

$$X_{i_1} \oplus X_{i_2} \oplus \ldots \oplus X_{i_u} \oplus Y_{j_1} \oplus Y_{j_2} \oplus \ldots \oplus Y_{j_v} = 0$$

- Korzystając z tej zależności próbujemy odtworzyć klucz.

ATAK RÓŻNICOWY

- Atakujący wybiera pary (plaintext, ciphertext)
- Patrzymy na różnicę między parami plaintekstu i jak przekłada się to na parę odpowiadających im zaszyfrowanych wiadomości
- W przypadku szyfru idealnego prawdopodobieństwo, że jedna różnica przełoży się na drugą to będzie $1/2^n$ (n liczba bitów).
- W ten sposób próbujemy odzyskiwać klucze rund

PYTANIA?

BIBLIOGRAFIA

- Prywatne materiały ze studiów
- https://www.engr.mun.ca/~howard/PAPERS/ldc_tutorial.pdf
- Wikipedia
- http://elartu.tntu.edu.ua/bitstream/123456789/1449/13/W_8_(SK_B)_uzup.pdf