

# Laboratorium z kryptografii

## Zajęcia 2: Szyfr S-DES

### 1 Zasada działania algorytmu

Jest to uproszczona wersja systemu DES, systemu korzystającego z szyfrowania z kluczem symetrycznym. Do odszyfrowania używa się tego samego klucza co do szyfrowania.

#### 1.1 Struktura algorytmu

Szyfr S-DES operuje na 8 bitowych blokach tekstu. W przypadku kiedy ostatnio blok tekstu posiada mniej niż 8 bitów, należy uzupełnić go zerami. **Przykładowo**  $(1, 1, 1, 1, 0, 0, 0) \rightarrow (1, 1, 1, 1, 0, 0, 0, 0)$ . Tak przygotowane bloki podlegają kolejnym krokom algorytmu:

1. Permutacja wstępna PW:

$$PW \equiv \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 1 & 5 & 2 & 0 & 3 & 7 & 4 & 6 \end{pmatrix}.$$

2. Szyfrowanie kluczem rundy pierwszej.
3. Krzyżowanie, czyli zamiana miejscami pierwszych 4 bitów z ostatnimi czterema, co można zapisać jako permutację:

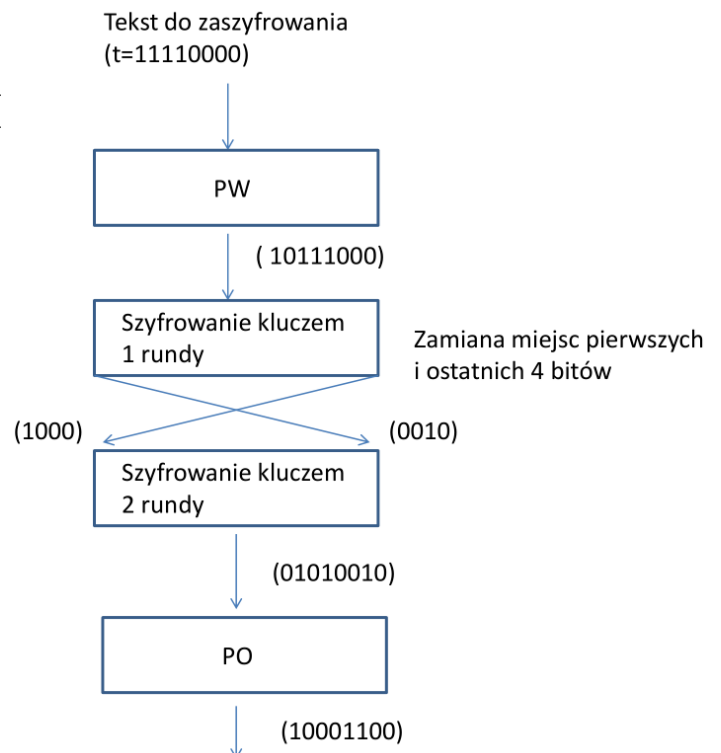
$$\begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 4 & 5 & 6 & 7 & 0 & 1 & 2 & 3 \end{pmatrix}.$$

4. Szyfrowanie kluczem rundy drugiej.
5. Algorytm kończy permutacja odwrotna PO:

$$PO \equiv \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 3 & 0 & 2 & 4 & 6 & 1 & 7 & 5 \end{pmatrix}.$$

Deszyfrowanie przebiega w kolejności:

1. Permutacja wstępna PW.
2. Szyfrowanie kluczem rundy drugiej.
3. Krzyżowanie.
4. Szyfrowanie kluczem rundy pierwszej.
5. Permutacja odwrotna PO.



Rysunek 1: Schemat blokowy algorytmu szyfrowania S-DES

## 1.2 Generacja kluczy pierwszej i drugiej rundy

Klucze pierwszej oraz drugiej rundy otrzymywane są na podstawie dziesięcio bitowego klucza początkowego  $k_p$  (przykładowo  $k_p = (1, 1, 0, 0, 0, 0, 0, 0, 1, 1)$ ) przetwarzanego w kolejnych krokach:

1. Permutacja P10 ciągu bitów  $k_p$  opisana wzorem:

$$P10 \equiv \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 2 & 4 & 1 & 6 & 3 & 9 & 0 & 8 & 7 & 5 \end{pmatrix}.$$

$$P10(k_p) = (0, 0, 1, 0, 0, 1, 1, 1, 0, 0).$$

2. Podział otrzymanego ciągu dziesięcio bitowego na dwa pięć bitowe  $k_0^0$  i  $k_0^1$  (na dwie połowy).

$$k_0^0 = (0, 0, 1, 0, 0), k_0^1 = (1, 1, 1, 0, 0).$$

3. Przesunięcie otrzymanych ciągów o jedną pozycję „w lewo” tj. permutacja SL1:

$$SL1 \equiv \begin{pmatrix} 0 & 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 & 0 \end{pmatrix}.$$

$$SL1(k_0^0) = (0, 1, 0, 0, 0), SL1(k_0^1) = (1, 1, 0, 0, 1).$$

4. Klucz rundy pierwszej otrzymuje się poprzez operację P10w8:

$$P10w8 \equiv \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 5 & 2 & 6 & 3 & 7 & 4 & 9 & 8 \end{pmatrix}.$$

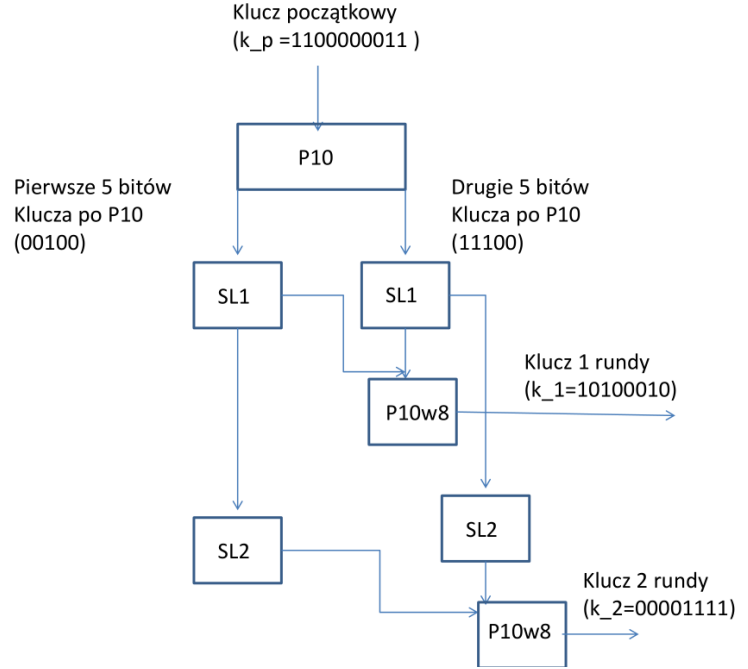
przenoszącą ciąg 10 bitowy powstały z połączenia otrzymanych w poprzednim kroku ciągów (w kolejności  $SL1(k_0^0) + SL1(k_0^1)$   $(0,1,0,0,0,1,1,0,0,1)$ ) na ciąg 8 bitowy.

$$k_1 = P10w8(0, 1, 0, 0, 0, 1, 1, 0, 0, 1) = (1, 0, 1, 0, 0, 0, 1, 0).$$

5. Klucz rundy drugiej rundy otrzymuje się poprzez operację P10w8 ciągu bitów  $SL2(SL1(k_0^0)) + SL2(SL1(k_0^1))$ , gdzie SL2

$$SL2 \equiv \begin{pmatrix} 0 & 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 0 & 1 \end{pmatrix}.$$

$$k_2 = PW10w8[SL2(SL1(k_0^0)) + SL2(SL1(k_0^1))] = PW10w8(0, 0, 0, 0, 1, 0, 0, 1, 1, 1) = (0, 0, 0, 0, 1, 1, 1, 1).$$



Rysunek 2: Schemat generacji kluczy I i II rundy

### 1.3 Szyfrowanie kluczem 1 i 2 rundy

Szyfrowanie kluczem 1 i 2 rundy przebiega w następującej kolejności

1. Tekst do zaszyfrowania  $t=(1,0,1,1,1,0,0,0)$  dzielony jest na dwa 4 bitowe ciągi  $(1,0,1,1)$ ,  $(1,0,0,0)$ . Z drugiego ciągu utworzona jest dodatkowa kopia.
2. Pierwsza z kopii drugiego ciągu  $(1,0,0,0)$  poddawana jest operacji P4w8:

$$P4w8 \equiv \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 3 & 0 & 1 & 2 & 1 & 2 & 3 & 0 \end{pmatrix}.$$

Tworzącą z ciągu 4 bitowego ciąg 8 bitowy.

$$P4w8(1,0,0,0) = (0,1,0,0,0,0,0,1).$$

3. Do otrzymanego ciągu dodawany jest binarnie (w ciele  $\mathbb{Z}_2$ ) klucz odpowiedniej rundy. W przypadku rundy pierwszej przykładowo otrzymuje się:

$$\begin{aligned} Xor[(0,1,0,0,0,0,0,1), (1,0,1,0,0,0,1,0)] &= \\ (0,1,0,0,0,0,0,1) \oplus (1,0,1,0,0,0,1,0) &= \\ (1,1,1,0,0,0,1,1). \end{aligned}$$

4. Wynik dodawania ponownie dzielony jest na dwa 4 bitowe ciągi. Pierwszy z nich przekształcany jest przez funkcje  $SBox1$  natomiast drugi przez  $SBox2$ , zdefiniowane części 1.4, tworząc dwa dwu bitowe ciągi.

$$SBox1(1,1,1,0) = (1,1) \quad SBox2(0,0,1,1) = (0,0).$$

5. Otrzymane ciągi łączone są w jeden ciąg 4 bitowy w kolejności wynik  $SBox1$  + wynik  $SBox2$ , po czym działa się na nie permutacją P4:

$$P4 \equiv \begin{pmatrix} 0 & 1 & 2 & 3 \\ 1 & 3 & 2 & 0 \end{pmatrix}.$$

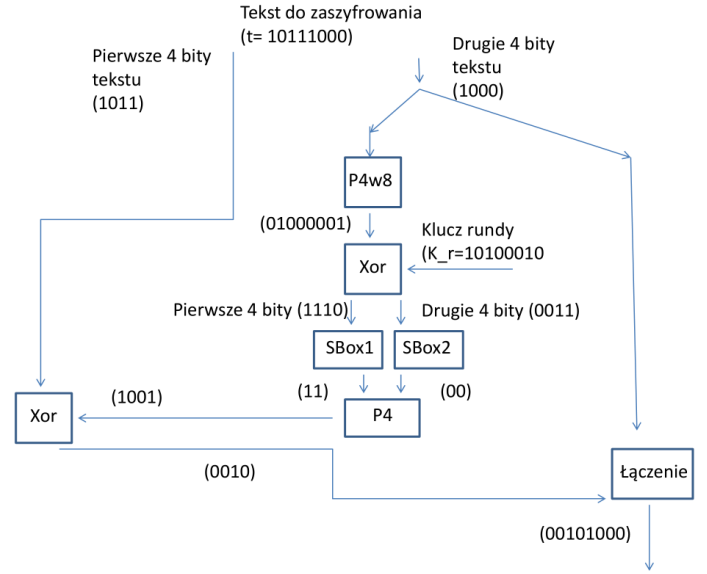
$$P4(1,1,0,0) = (1,0,0,1).$$

6. Uzyskany ciąg dodawany jest binarnie do pierwszych czterech bitów tekstu t.

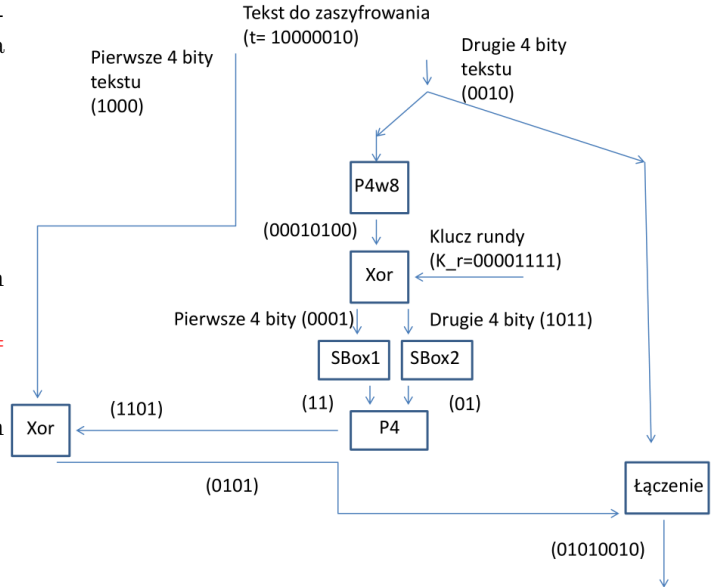
$$Xor[(1,0,1,1), (1,0,0,1)] = (1,0,1,1) \oplus (1,0,0,1) = (0,0,1,0).$$

7. Otrzymana suma łączona jest z drugą kopią drugich czterech bitów tekstu t tworząc 8 bitowy ciąg.

$$(0,0,1,0) + (1,0,0,0) = (0,0,1,0,1,0,0,0)$$



Rysunek 3: Szyfrowanie kluczem rundy pierwszej



Rysunek 4: Szyfrowanie kluczem rundy drugiej

Szyfrowanie kluczem rundy drugiej przebiega w sposób analogiczny przy użyciu klucza  $k_2$  w miejscu klucza  $k_1$  w omówionej porcedurze.

## 1.4 SBox1 i SBox2

SBoxy są to funkcje nieliniowe zapewniające bezpieczeństwo wielu nowoczesnych szyfrów. W omawianym przykładzie algorytmu S-DES *SBox1* oraz *SBox2* można przedstawić w postaci tabel:

<i>SBox1</i>	c0	c1	c2	c3
r0	1	0	3	2
r1	3	2	1	0
r2	0	2	1	3
r3	3	1	3	2

<i>SBox2</i>	c0	c1	c2	c3
r0	0	1	2	3
r1	2	0	1	3
r2	3	0	1	0
r3	2	1	0	3

Na podstawie 4 bitowego ciągu wejściowego  $(a, b, c, d)$ , gdzie  $a, b, c, d \in \{0, 1\}$ , *SBox* zwraca dwu bitowy ciąg kodujący jedną z liczb z tabeli. Aby określić wiersz, w którym znajduje się poszukiwana liczba, należy przekształcić ciąg złożony z pierwszego i czwartego bitu ciągu wejściowego (tj.  $(a, d)$ ) do postaci dziesiętnej. Kolumnę natomiast wyznacza drugi oraz trzeci wyraz ciągu wejściowego (tj. ciąg  $(b, c)$ ).

Przykład:

$SBox1(1, 1, 1, 0) \rightarrow$  wiersz:  $(1, 0) = 2$  kolumna:  $(1, 1) = 3 \Rightarrow SBox1(1, 1, 1, 0) = (1, 1)$   
 $SBox2(0, 0, 1, 1) \rightarrow$  wiersz:  $(0, 1) = 1$  kolumna:  $(0, 1) = 1 \Rightarrow SBox2(0, 0, 1, 1) = (0, 0)$

## 2 ZADANIA

1. Dla zadanego z konsoli ciągu bitów (maksymalnie długości 8 bitów) oraz klucza długości dokładnie 10 bitów (może być ustawiony „na sztywno” w programie) napisać program szyfrujący algorytmem S-DES
2. Dla zadanego z konsoli szyfrogramu (maksymalnie długości 8 bitów) oraz klucza długości dokładnie 10 bitów (może być ustawiony „na sztywno” w programie) napisać program deszyfrujący algorytmem S-DES.

Punktacja - łącznie 10 punktów

- 1 punkt - wczytywanie tekstu (o długości do 8 bitów) i uzupełnianie go zerami w przypadku ciągu krótszego niż 8 znaków.
- 3 punkty - poprawna generacja kluczy I i II rundy oraz wyświetlenie ich w programie.
- 2 punkty - poprawna implementacja *SBoxów* i zwracanych przez nie wartości w programie w systemie dziesiętnym.
- 3 punkty - poprawna implementacja szyfrowania kluczem I i II rundy oraz wyświetlenie ich wyników w programie
- 1 punkt - poprawne uzyskiwanie szyfrogramu i jego odszyfrowywanie (wyświetlone w programie)