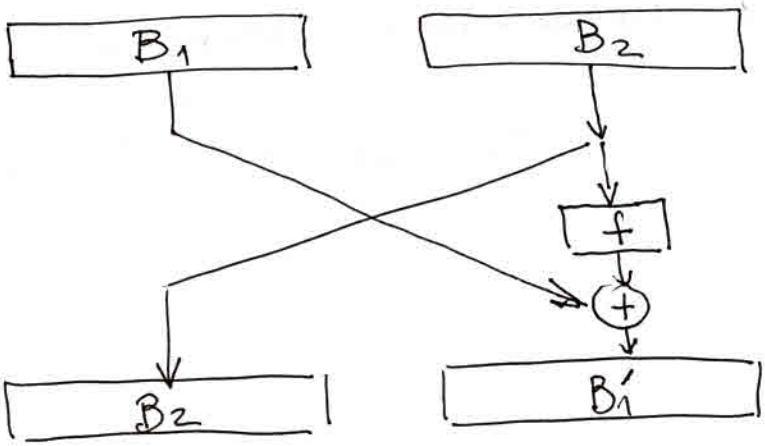
Шифри на Файстел

- Размерът на блоковете прави да е голям: защитата за блок
- Всеки блок открит текст се разделя на две части: $m = (\underbrace{m_0}_{\text{n-бита}}, \underbrace{m_1}_{\text{n-бита}})$
- Ключът k определя множество от преключове:
$$k_1, k_2, \ldots, k_h$$
за някакво цяло число h
- За всяко k_i съответства трансформация f_{k_i}, която преобразува множеството от блокове с дължина n в друго себе си
$$f_{k_i}: \{0,1\}^n \to \{0,1\}^n$$
- Всяко съобщение се шифрова в n стъпки:



$$R' = R \oplus f(B_2)$$

$$\mu_0 = (m_0, m_1) \to \mu_1 = (m_1, m_2)$$
$$\mu_1 = (m_1, m_2) \to \mu_2 = (m_2, m_3)$$
$$\vdots$$
$$\mu_{i-1} = (m_{i-1}, m_i) \to \mu_i = (m_i, m_{i+1})$$
$$\vdots$$
$$\mu_{k-1} = (m_{k-1}, m_k) \to \mu_k = (m_k, m_{k+1})$$
$$\downarrow$$
$$\overline{\mu_k} = (m_{k+1}, m_k)$$

По-горе: $m_{i+1} = m_{i-1} + f_{k_i}(m_i)$

Да отбележим, че

$$m_{i-1} = m_{i+1} + f_{k_i}(m_i)$$

Това позволява да дешифрираме
като приложим процедурата за шифриране
с поредността взета в обратен ред:

$$k_k, k_{k-1}, \ldots k_2, k_1$$

Дешифриране:

$$\bar{\mu}_h = (m_{h+1}, m_h) \to \bar{\mu}_{h-1} = (m_h, m_{h-1}) \qquad k_h$$

$$\vdots$$

$$\bar{\mu}_i = (m_{i+1}, m_i) \to \bar{\mu}_{i-1} = (m_i, m_{i-1}) \qquad k_i$$

$$\vdots$$

$$\bar{\mu}_1 = (m_2, m_1) \to \bar{\mu}_0 = (m_1, m_0) \qquad k_1$$

$$\downarrow$$

$$\bar{\bar{\mu}}_0 = \mu_0 = (m_0, m_1)$$

$$m_{i-1} = m_{i+1} + f_{k_i}(m_i)$$

Пример.

Нека $n = 2$, $h = 4$

	00	01	10	11
f_{k_1}	10	10	00	01
f_{k_2}	01	11	01	10
f_{k_3}	10	11	00	01
f_{k_4}	11	10	01	00

Шифриране

0010 → 1000
1000 → 0011
0011 → 1101
1101 → 0101

↓

0101

Дешифриране

0101 → 0111
0111 → 1100
1100 → 0010
0010 → 1000

$\boxed{m = 0010}$

00 + 00 = 00
10 + 01 = 11
00 + 01 = 01
11 + 10 = 01

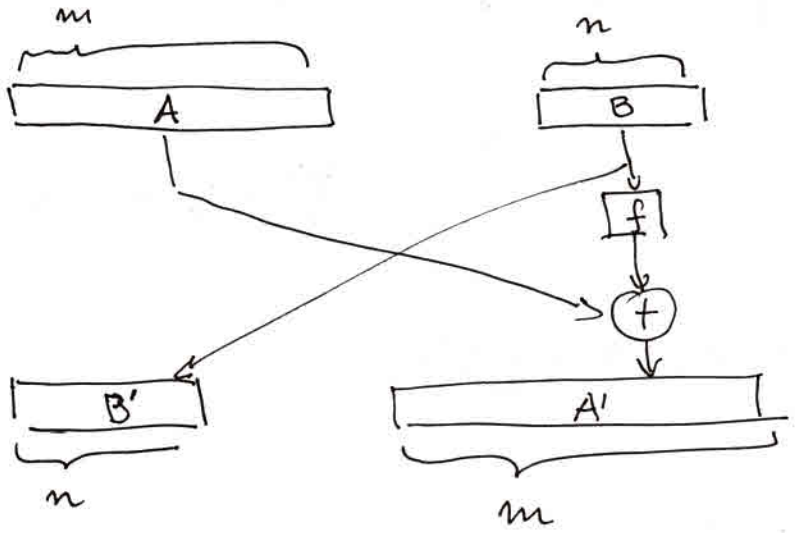
01 + 10 = 11
01 + 01 = 00
11 + 01 = 10
00 + 00 = 00

Механизм Фейстеля

A – с длиной m
B – с длиной n

$$(\underbrace{A}_{m}, \underbrace{B}_{n}) \rightarrow (\underbrace{A'}_{n}, \underbrace{B'}_{m})$$

$$\begin{cases} A' = B \\ B' = A + f_i(B) \end{cases} \qquad f_i : \{0,1\}^n \rightarrow \{0,1\}^m$$



Используется в Skipjack.