**Молдавский государственный университет**

**Факультет математики и информатики**

**Департамент Информатики**

**КРИПТОАНАЛИЗ И КРИПТОГРАФИЧЕСКИЕ ПРОТОКОЛЫ**

**ATESTARE №1**

Выполнена студентом III курса

специальности Прикладная Информатика

Гуцу Даниил

Проверила Ольга Чербу, doctor conferențiar universitar

**КИШИНЕВ – 2023**

ОГЛАВЛЕНИЕ

[1. ОПИСАНИЕ СЕТИ ФЕЙСТЕЛЯ 3](#_Toc146964319)

[2. АЛГОРИТМ RC2 4](#_Toc146964320)

[2.1. Шифрование 4](#_Toc146964321)

[2.2. Дешифрование 4](#_Toc146964322)

[3. АЛГОРИТМ TEA 5](#_Toc146964323)

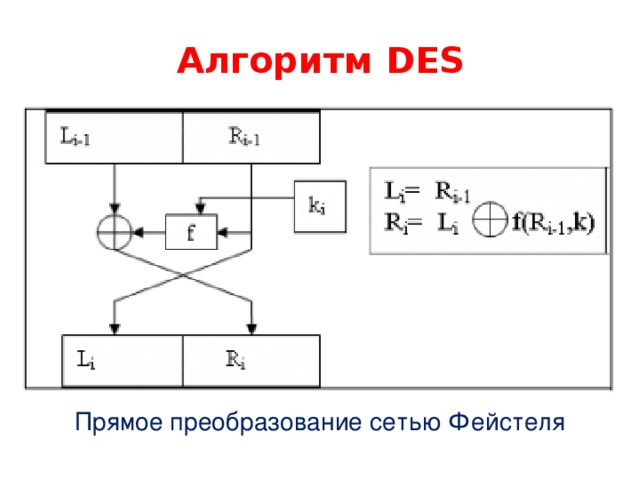
[3.1. Шифрование 5](#_Toc146964324)

[3.2. Дешифрование 5](#_Toc146964325)

# ОПИСАНИЕ СЕТИ ФЕЙСТЕЛЯ

**Сеть Фе́йстеля** (конструкция Фейстеля) — один из методов построения блочных шифров. Сеть представляет собой определённую многократно повторяющуюся (иттерированную) структуру, называющуюся ячейкой Фейстеля. При переходе от одной ячейки к другой меняется ключ, причём выбор ключа зависит от конкретного алгоритма. Операции шифрования и расшифрования на каждом этапе очень просты, и при определённой доработке совпадают, требуя только обратного порядка используемых ключей. Шифрование при помощи данной конструкции легко реализуется как на программном уровне, так и на аппаратном, что обеспечивает широкие возможности применения. Большинство современных блочных шифров используют сеть Фейстеля в качестве основы. Альтернативой сети Фейстеля является подстановочно – перестановочная сеть.

Примеры сети Фейстеля:

****

# АЛГОРИТМ RC2

Выполнить шифрование и дешифрование алгоритмом RC2, используемом в протоколах сетей Wi-Fi. Выберите сообщение, которое желаете, не превышающее длину 8 символов(criptogg). Ключ будет состоять из инициалов студента: (**Пример: Gutu Daniil. Ключ: GD**).

Изображение выглядит как диаграмма, линия, План, Технический чертеж

Автоматически созданное описание

## Шифрование

01100011 01110010 01101001 01110000

01110100 01101111 01100111 01100111

a = 01100011 01110010

b = 01101001 01110000

c = 01110100 01101111

d = 01100111 01100111

Subkey = GD = 01000111 01000100

Step 1: a + key = 43702 = 10101010 10110110

Step 2: c ^ d = 01100100 01100111

Step 3: (not d) ^ b = 0000100000010000

Step 4: (c ^ d) + ((not d) ^ b) = 01101100 01110111

Step 5: a + key + (c ^ d) + ((not d) ^ b) = 100010111 00101101

Step 6: part5 <<< 4 = 000010001 01110010 11010000

Step 7: b => 0110100101110000

c => 0111010001101111

d => 0110011101100111

a => 00000000000100010111001011010000

Step 8: Concatenating b, c, d, a => 01101001 01110000 01110100 01101111 01100111 01100111 00000000 00010001 01110010 11010000

Result = "iptogg" + "1172d0"

## Дешифрование

Step 1: part6 >>> 4 = 100010111 00101101

Step 2: c ^ d = 01100100 01100111

Step 3: (not d) ^ b = 00001000 00010000

Step 4: (c ^ d) + ((not d) ^ b) = 01101100 01110111

Step 5: (a - (b ^ (not d) + (c ^ d) )) => 43702 = 10101010 10110110

Step 6: part5 - CV => 25458 = 0110001101110010

Result = "criptogg"

# АЛГОРИТМ TEA

Выполнить шифрование и дешифрование алгоритмом TEA. Каждый студент выбирает сообщение длиной в 8 символов(criptogg). В ключ K[0] вводятся инициалы как в примере: **gd** – Gutu Daniil.

## Шифрование

Message: criprogg

Message in binary: 01100011 01110010 01101001 01110000 01110010 01101111 01100111 01100111

Left side - 01100011 01110010 01101001 01110000

Right side - 01110010 01101111 01100111 01100111

Keys: [gdgr, adua, teun, iver]

K0: [01100111, 01100100, 01100111, 01110010]

K1: [01100001, 01100100, 01110101, 01100001]

K2: [01110100, 01100101, 01110101, 01101110]

K3: [01101001, 01110110, 01100101, 01110010]

1: right << 4

01011101 01111110 00110101 11001110 << 4

11010111 11100011 01011100 11100000

2: ((1) + K2) mod 2^32

11010111 11100011 01011100 11100000 +

01110100 01100101 01110101 01101110 =

01001100 01001000 11010010 01001110

3: (right + DELTA) mod 2^32

01011101 01111110 00110101 11001110 +

10011110 00110111 01111001 10111001 =

11111011 10110101 10101111 10000111

4: (2) xor (3)

01001100 01001000 11010010 01001110 xor

11111011 10110101 10101111 10000111 =

10110111 11111101 01111101 11001001

5: right >> 5

01011101 01111110 00110101 11001110 >> 5

00000010 11101011 11110001 10101110

6: ((5) + K3) mod 2^32

00000010 11101011 11110001 10101110 +

01101001 01110110 01100101 01110010 =

01101100 01100010 01010111 00100000

7: (4) xor (6)

10110111 11111101 01111101 11001001 xor

01101100 01100010 01010111 00100000 =

11011011 10011111 00101010 11101001

8: (left + (7)) mod 2^32

01110010 01101111 01100111 01100111 +

11011011 10011111 00101010 11101001 =

01001110 00001110 10010010 01010000

01011101 01111110 00110101 11001110 01001110 00001110 10010010 01010000

Result:

01011101 01111110 00110101 11001110 01001110 00001110 10010010 01010000,

]嵾縵㗎칎与ຒ鉐

1: right << 4

01011101 01111110 00110101 11001110 << 4

11010111 11100011 01011100 11100000

2: ((1) + K2) mod 2^32

11010111 11100011 01011100 11100000 +

01110100 01100101 01110101 01101110 =

01001100 01001000 11010010 01001110

3: (right + DELTA) mod 2^32

01011101 01111110 00110101 11001110 +

10011110 00110111 01111001 10111001 =

11111011 10110101 10101111 10000111

4: (2) xor (3)

01001100 01001000 11010010 01001110 xor

11111011 10110101 10101111 10000111 =

10110111 11111101 01111101 11001001

5: right >> 5

01011101 01111110 00110101 11001110 >> 5

00000010 11101011 11110001 10101110

6: ((5) + K3) mod 2^32

00000010 11101011 11110001 10101110 +

01101001 01110110 01100101 01110010 =

01101100 01100010 01010111 00100000

7: (4) xor (6)

10110111 11111101 01111101 11001001 xor

01101100 01100010 01010111 00100000 =

11011011 10011111 00101010 11101001

8: (left - (7)) mod 2^32

01001110 00001110 10010010 01010000 -

11011011 10011111 00101010 11101001 =

01110010 01101111 01100111 01100111

## Дешифрование

1: right << 4

01011101 01111110 00110101 11001110 << 4

11010111 11100011 01011100 11100000

2: ((1) + K2) mod 2^32

11010111 11100011 01011100 11100000 +

01110100 01100101 01110101 01101110 =

01001100 01001000 11010010 01001110

3: (right + DELTA) mod 2^32

01011101 01111110 00110101 11001110 +

10011110 00110111 01111001 10111001 =

11111011 10110101 10101111 10000111

4: (2) xor (3)

01001100 01001000 11010010 01001110 xor

11111011 10110101 10101111 10000111 =

10110111 11111101 01111101 11001001

5: right >> 5

01011101 01111110 00110101 11001110 >> 5

00000010 11101011 11110001 10101110

6: ((5) + K3) mod 2^32

00000010 11101011 11110001 10101110 +

01101001 01110110 01100101 01110010 =

01101100 01100010 01010111 00100000

7: (4) xor (6)

10110111 11111101 01111101 11001001 xor

01101100 01100010 01010111 00100000 =

11011011 10011111 00101010 11101001

8: (left - (7)) mod 2^32

01110010 01101111 01100111 01100111 -

11011011 10011111 00101010 11101001 =

10010110 11010000 00111100 01111110

Decryption:

01100011 01110010 01101001 01110000 01110010 01101111 01100111 01100111

Message: criprogg