Университет ИТМО
Факультет ПИиКТ
Информационная безопасность

Лабораторная работа №2
«Блочное симметричное шифрование»

Вербовой Александр
Группа P3400
Вариант 4

Санкт-Петербург
2021

Цель работы

изучение структуры и основных принципов работы современных алгоритмов блочного симметричного шифрования, приобретение навыков программной реализации блочных симметричных шифров.

Задание (вариант 4)

Алгоритм – ГОСТ 28147-89. Режим шифрования - CFB

Описание:

```
sbox = (
        (4, 10, 9, 2, 13, 8, 0, 14, 6, 11, 1, 12, 7, 15, 5, 3),
        (14, 11, 4, 12, 6, 13, 15, 10, 2, 3, 8, 1, 0, 7, 5, 9),
        (5, 8, 1, 13, 10, 3, 4, 2, 14, 15, 12, 7, 6, 0, 9, 11),
        (7, 13, 10, 1, 0, 8, 9, 15, 14, 4, 6, 12, 11, 2, 5, 3),
        (6, 12, 7, 1, 5, 15, 13, 8, 4, 10, 9, 14, 0, 3, 11, 2),
        (4, 11, 10, 0, 7, 2, 1, 13, 3, 6, 8, 5, 9, 12, 15, 14),
        (13, 11, 4, 1, 3, 15, 5, 9, 0, 10, 14, 7, 6, 8, 2, 12),
        (1, 15, 13, 0, 5, 7, 10, 4, 9, 2, 3, 14, 6, 11, 8, 12),
        )
text = 0xdeadcafebeda0313
key = 19313867402948313029866648204957124098313134590881238590123857510238313108576
```

```
def encrypt(msg, key):
    # Открытый текст сначала разбивается на две половины
    # старшие биты — left_part, младшие биты — right_part
    left_part = msg >> 32
    right_part = msg & 0xFFFFFFFF
    # Для генерации подключей исходный 256-битный ключ разбивается на восемь 32-битных блоков: K1…K8.
    subkeys = [(key >> (32 * i)) & 0xFFFFFFFF for i in range(8)]
    # Выполняем 32 рауда со своим подключом Ki
    # Ключи K1…K24 являются циклическим повторением ключей K1…K8 (нумеруются от младших битов к старшим).
    for i in range(24):
        left_part, right_part = encrypt_round_cfb(left_part, right_part, subkeys[i % 8])
    for i in range(8):
        left_part, right_part = encrypt_round_cfb(left_part, right_part, subkeys[7 - i])
    return (left_part << 32) | right_part # сливаем половинки вместе
```

```
def encrypt_round_cfb(left_part, right_part, round_key):
    return right_part, left_part ^ f(right_part, round_key) # xor и смена их местами
```

```
def f(part, key):
    temp = part ^ key #Складываем по модулю
    output = 0
    # Разбиваем по 4бита
    # в результате sbox[i][j] где i-номер шага, j-значение 4битного куска i шага
    # выходы всех восьми S-блоков объединяются в 32-битное слово
    for i in range(8):
        output |= ((sbox[i][(temp >> (4 * i)) & 0b1111]) << (4 * i))
    # всё слово циклически сдвигается влево (к старшим разрядам) на 11 битов.
    return ((output << 11) | (output >> (32 - 11))) & 0xFFFFFFFF
```

```python
def f(part, key):
    temp = part ^ key #Складываем по модулю
    output = 0
    # Разбиваем по 4бита
    # в результате sbox[i][j] где i-номер шага, j-значение 4битного куска i шага
    # выходы всех восьми S-блоков объединяются в 32-битное слово
    for i in range(8):
        output |= ((sbox[i][(temp >> (4 * i)) & 0b1111]) << (4 * i))
    # всё слово циклически сдвигается влево (к старшим разрядам) на 11 битов.
    return ((output << 11) | (output >> (32 - 11))) & 0xFFFFFFFF
```

```python
def decrypt(msg, key):
    # текст сначала разбивается на две половины
    # старшие биты — left_part, младшие биты — right_part
    left_part = msg >> 32
    right_part = msg & 0xFFFFFFFF
    # Для генерации подключей исходный 256-битный ключ разбивается на восемь 32-битных блоков: K1…K8.
    subkeys = [(key >> (32 * i)) & 0xFFFFFFFF for i in range(8)]
    # Выполняем 32 рауда со своим подключом Ki
    # Ключи K1…K24 являются циклическим повторением ключей K1…K8 (нумеруются от младших битов к старшим).
    # В порядке обратному порядку в шифровании
    for i in range(8):
        left_part, right_part = decrypt_round_cfb(left_part, right_part, subkeys[i])
    for i in range(24):
        left_part, right_part = decrypt_round_cfb(left_part, right_part, subkeys[(7 - i) % 8])
    return (left_part << 32) | right_part #сливаем половинки вместе
```

```python
def decrypt_round_cfb(left_part, right_part, round_key):
    return right_part ^ f(left_part, round_key), left_part # xor и смена их местами
```

```python
encrypted = encrypt(text, key)
decrypted = decrypt(encrypted, key)
text, encrypted, decrypted
```

```
(16045704242863407891, 1921941495089229986, 16045704242863407891)
```

Вывод

В ходе выполнения лабораторной работы были изучены структуры и основные принципы работы современных алгоритмов блочного симметричного шифрования.