Универзитет „Св. Кирил и Методиј“ во Скопје
ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

Домашна задача бр.3
Криптографија

Блок шифрувач - DES

Стефан Андонов, 151020
stefan.andonov@students.finki.ukim.mk

Во оваа домашна задача ќе прикажам имплементација на блок шифрувачот DES, имплементирана во програмскиот јазик Java. Имплементацијата е поставена на GitHub. Истата е реализирана со помош на 4 класи:

- RoundKeyGenerator.java
  Со помош на оваа класа, генерираме 56 битни клучеви кои што ќе се користат во секоја рунда, од еден главен 64 битен клуч
- FeistelFunction.java
  Со помош на оваа класа, се извршува Феистеловата функција во рамки на една рунда, односно се прави проширување, пермутација и намалување со помош на 8 различни S-box-ови.
- DES.java
  Во оваа класа креираме објекти од претходните две класи (RoundKeyGenerator и FeistelFunction) кои ќе ги користиме за дефинирање на функции во класата, со цел да извршиме енкрипција и декрипција на податоците.
- DESTester.java
  Во оваа класа ќе ја тестираме имплентацијата на претходните три наведени класи, и ќе ја искористиме истата за да ги реализираме сите три барања кои што се зададени во домашната задача.

Задача 1. Да се направи имплементација на блок шифрувачот DES во програмски јазик по желба и да се провери дали соодветно работат енкрипцијата и декрипцијата на дадена порака M со ист клуч K.

Ја шифираме пораката “KRIPTOGR”, која што кога се претвара во битови ќе има должина од 64 бита, колку што треба да биде еден блок во DES шифрувачот. За клуч случајно го избираме да биде бројот 12345678999.

```
String M = "KRIPTOGR";

//1.
long Mencrypted = des.encrypt(messageToLong(M),1234568999);
String C = longToMessage(Mencrypted);
String Cbits = Long.toBinaryString(Mencrypted);
System.out.println(C);
System.out.println(Cbits);
```

```
            long Cdecrypted = des.decrypt(messageToLong(C),1234568999);
            String M1 = longToMessage(Cdecrypted);
            System.out.println(M1);
            System.out.println(Long.toBinaryString(Cdecrypted));
```

Функциите messageToLong(String M) и longToMessage(long l) се дефинирани со цел да можат да претвораат еден стринг од 8 карактери во блок од 64 бита, и обратното. Излезот што го добиваме со извршување на кодниот сегмент погоре е следниот:

```
_SY_UJ__
0000101110101001101011001000001010101010101001010000100110000011 0
KRIPTOGR
0100101110101001001001001010100000101010001001111010001110101001 0
```

Од ова може да заклучиме дека алгоритмот е функционален, односно работи и енкрипцијата и декрипцијата.

а) Кој е излезот од првата рунда на DES кога и пораката и клучот се составени само од нули?

```
//2
            DES.NUM_OF_ROUNDS=1;
            long M = 0;
            long K = 0;
            System.out.println(Long.toBinaryString(des.encrypt(M, K)));
```

Излезот е следниот:

```
0000010000000100000000010101010101010101000000010101010001010101
0000010000000100000000010101010101010101000000010101010001010100
```

б)Кој е излезот од првата рунда на DES кога и пораката и клучот се составени само од единици?

```
//3
            long M = Long.MAX_VALUE;
            long K = Long.MAX_VALUE;
            System.out.println(Long.toBinaryString(des.encrypt(M, K)));
```

```
0110101111111011101111111011101010101111111111101111101110101010
```

в) Кој е излезот од првата рунда на DES ако пораката од а) се променила во еден бит, а клучот останува да биде со сите нули?

```
//4
            long M = 1;
            long K = 0;
            System.out.println(Long.toBinaryString(des.encrypt(M, K)));
0000010000000100000000010101010101010101000000010101010001010100
```

г) Колку битови се променети на излез, со промена на едниот бит споредено со резултатот под а)? Има ли разлика? Колкав е авеланч ефектот тука?

Има разлика во само еден бит (последниот бит). За да има авеланч ефект потребно е да се променат барем 50% од битовите, но тој не е постигнат од причина што извршуваме само една рунда на DES алгоритмот, а тоа не е доволно.

д) Споредете колку од S-box-овите сега ќе имаат различни влезови (во втората рунда), споредено со ситуацијата под а)

За да го процениме ова ќе биде потребно да правиме печатење секогаш кога се врши замената во S-box-овите, односно да го печатиме бројот на S-box-от, двата влеза и излезите. Ова ќе го постигнеме со додавање на една линија код во функцијата `private byte SBoxSubstitution(int SBoxNum, byte input)`, во класата FeistelFunction.

```
private byte SBoxSubstitution(int SBoxNum, byte input)
    {
       byte row, col;
       row = (byte) (((input & 0x20) >> 4) | input & 0x01);
       col = (byte) ((input & 0x1E) >> 1);
       System.out.println(String.format("SBOX %d [%d][%d] = %d\n", SBoxNum, row, col,
S_BOX[SBoxNum][row][col])); //izmeneto
         return S_BOX[SBoxNum][row][col];
    }
```

Исто така, ќе вметниме и една линија код во функцијата `DES.cipher(long block, long key, boolean encrypt)` за да се печати бројот на рунда.

Кодот којшто ќе биде во класата DESTester за да ги добиеме посакуваните резултати е следниот:

```
       //d)
              DES.NUM_OF_ROUNDS=2;
              long M=0;
              long M1=1;
              long K=0;
              System.out.println(Long.toBinaryString(des.encrypt(M, K)));
              System.out.println(Long.toBinaryString(des.encrypt(M1, K)));
```

Со повикување на функцијата encrypt, се повикуваат и функциите кои што погоре ги изменивме.

Резултатот којшто се добива е следниот:

M = 0…000 K= 0…000	M=0…001 K=0…001
Runda 1 SBOX 0 [0][0] = 14 SBOX 1 [0][0] = 15 SBOX 2 [0][0] = 10 SBOX 3 [0][0] = 7 SBOX 4 [0][0] = 2 SBOX 5 [0][0] = 12 SBOX 6 [0][0] = 4 SBOX 7 [0][0] = 13	Runda 1 SBOX 0 [0][0] = 14 SBOX 1 [0][0] = 15 SBOX 2 [0][0] = 10 SBOX 3 [0][0] = 7 SBOX 4 [0][0] = 2 SBOX 5 [0][0] = 12 SBOX 6 [0][0] = 4 SBOX 7 [0][0] = 13

Runda 2 SBOX 0 [1][13] = 5 SBOX 1 [3][8] = 11 SBOX 2 [1][13] = 11 SBOX 3 [3][8] = 9 SBOX 4 [1][13] = 9 **SBOX 5 [3][11] = 7** **SBOX 6 [3][11] = 15** SBOX 7 [3][12] = 3	Runda 2 SBOX 0 [1][13] = 5 SBOX 1 [3][8] = 11 SBOX 2 [1][13] = 11 SBOX 3 [3][8] = 9 SBOX 4 [1][13] = 9 **SBOX 5 [2][11] = 10** **SBOX 6 [3][3] = 8** SBOX 7 [3][12] = 3

Може јасно да се согледа дека во првата рунда, всушност нема промена ниту во влезот на s-box-овите, ниту пак во излезот. Додека пак, во втората рунда имаме промена во влезот и излезот на два S-box-а, петтиот и шестиот.

ѓ) Кои S-box-ови се засегнати од промената на битот? Да се прикаже патот на промената во сите 16 рунди?

За да го добиеме овој резултат потребно е да го зголемиме бројот на рунди на 16.

M = 0…000 K= 0…000	M=0…001 K=0…001
Runda 1 SBOX 1 [0][0] = 14 SBOX 2 [0][0] = 15 SBOX 3 [0][0] = 10 SBOX 4 [0][0] = 7 SBOX 5 [0][0] = 2 SBOX 6 [0][0] = 12 SBOX 7 [0][0] = 4 SBOX 8 [0][0] = 13	Runda 1 SBOX 1 [0][0] = 14 SBOX 2 [0][0] = 15 SBOX 3 [0][0] = 10 SBOX 4 [0][0] = 7 SBOX 5 [0][0] = 2 SBOX 6 [0][0] = 12 SBOX 7 [0][0] = 4 SBOX 8 [0][0] = 13
Runda 2 SBOX 1 [1][13] = 5 SBOX 2 [3][8] = 11 SBOX 3 [1][13] = 11 SBOX 4 [3][8] = 9 SBOX 5 [1][13] = 9 SBOX 6 [3][11] = 7 SBOX 7 [3][11] = 15 SBOX 8 [3][12] = 3	Runda 2 SBOX 1 [1][13] = 5 SBOX 2 [3][8] = 11 SBOX 3 [1][13] = 11 SBOX 4 [3][8] = 9 SBOX 5 [1][13] = 9 SBOX 6 [2][11] = 10 SBOX 7 [3][3] = 8 SBOX 8 [3][12] = 3
Runda 3 SBOX 1 [2][14] = 5 SBOX 2 [0][7] = 4 SBOX 3 [3][3] = 0 SBOX 4 [3][10] = 5 SBOX 5 [1][14] = 8 SBOX 6 [0][13] = 7 SBOX 7 [3][4] = 1	Runda 3 SBOX 1 [2][15] = 0 SBOX 2 [2][5] = 4 SBOX 3 [3][2] = 13 SBOX 4 [1][10] = 2 SBOX 5 [1][12] = 3 SBOX 6 [0][9] = 13 SBOX 7 [2][4] = 12

SBOX 8 [1][15] = 2	SBOX 8 [1][7] = 4
Runda 4 SBOX 1 [1][5] = 2 SBOX 2 [3][11] = 12 SBOX 3 [3][15] = 12 SBOX 4 [2][10] = 3 SBOX 5 [1][6] = 13 SBOX 6 [1][11] = 14 SBOX 7 [2][10] = 6 SBOX 8 [0][6] = 11	Runda 4 SBOX 1 [3][14] = 6 SBOX 2 [1][12] = 6 SBOX 3 [1][8] = 2 SBOX 4 [1][9] = 7 SBOX 5 [3][15] = 3 SBOX 6 [3][10] = 1 SBOX 7 [0][8] = 3 SBOX 8 [1][5] = 3
Runda 5 SBOX 1 [3][5] = 9 SBOX 2 [2][14] = 2 SBOX 3 [0][4] = 6 SBOX 4 [1][5] = 15 SBOX 5 [2][14] = 0 SBOX 6 [0][2] = 10 SBOX 7 [0][5] = 0 SBOX 8 [2][7] = 2	Runda 5 SBOX 1 [2][1] = 1 SBOX 2 [2][7] = 1 SBOX 3 [3][6] = 8 SBOX 4 [0][8] = 1 SBOX 5 [0][7] = 6 SBOX 6 [3][7] = 10 SBOX 7 [2][13] = 5 SBOX 8 [2][3] = 1
Runda 6 SBOX 1 [1][8] = 10 SBOX 2 [0][11] = 13 SBOX 3 [3][1] = 10 SBOX 4 [2][9] = 1 SBOX 5 [3][7] = 13 SBOX 6 [3][11] = 7 SBOX 7 [2][14] = 9 SBOX 8 [1][0] = 1	Runda 6 SBOX 1 [3][7] = 7 SBOX 2 [3][14] = 14 SBOX 3 [1][11] = 14 SBOX 4 [3][13] = 7 SBOX 5 [2][11] = 5 SBOX 6 [2][3] = 5 SBOX 7 [2][0] = 1 SBOX 8 [0][7] = 1
Runda 7 SBOX 1 [1][15] = 8 SBOX 2 [3][13] = 5 SBOX 3 [3][14] = 2 SBOX 4 [1][9] = 7 SBOX 5 [3][8] = 6 SBOX 6 [0][9] = 13 SBOX 7 [3][4] = 1 SBOX 8 [1][10] = 6	Runda 7 SBOX 1 [2][15] = 0 SBOX 2 [2][5] = 4 SBOX 3 [2][2] = 4 SBOX 4 [1][5] = 15 SBOX 5 [3][12] = 10 SBOX 6 [1][12] = 0 SBOX 7 [1][12] = 2 SBOX 8 [1][13] = 14
Runda 8 SBOX 1 [1][1] = 15 SBOX 2 [3][9] = 6 SBOX 3 [3][13] = 5 SBOX 4 [2][15] = 4 SBOX 5 [3][0] = 11 SBOX 6 [0][11] = 4 SBOX 7 [3][5] = 4 SBOX 8 [2][12] = 15	Runda 8 SBOX 1 [2][15] = 0 SBOX 2 [3][7] = 2 SBOX 3 [3][15] = 12 SBOX 4 [3][14] = 2 SBOX 5 [0][10] = 3 SBOX 6 [1][7] = 5 SBOX 7 [2][15] = 2 SBOX 8 [3][7] = 13
Runda 9 SBOX 1 [0][9] = 10 SBOX 2 [2][0] = 0 SBOX 3 [1][7] = 10 SBOX 4 [2][10] = 3 SBOX 5 [0][1] = 12 SBOX 6 [3][3] = 12 SBOX 7 [2][15] = 2 SBOX 8 [3][0] = 2	Runda 9 SBOX 1 [3][9] = 11 SBOX 2 [2][13] = 3 SBOX 3 [2][6] = 3 SBOX 4 [1][4] = 6 SBOX 5 [0][14] = 14 SBOX 6 [0][1] = 1 SBOX 7 [2][6] = 7 SBOX 8 [1][5] = 3

Runda 10 SBOX 1 [1][8] = 10 SBOX 2 [0][8] = 9 SBOX 3 [1][1] = 7 SBOX 4 [2][9] = 1 SBOX 5 [3][0] = 11 SBOX 6 [0][12] = 14 SBOX 7 [0][5] = 0 SBOX 8 [3][0] = 2	Runda 10 SBOX 1 [2][11] = 7 SBOX 2 [2][0] = 0 SBOX 3 [1][2] = 0 SBOX 4 [1][8] = 4 SBOX 5 [1][13] = 9 SBOX 6 [2][9] = 0 SBOX 7 [2][7] = 14 SBOX 8 [3][1] = 1
Runda 11 SBOX 1 [0][2] = 13 SBOX 2 [1][5] = 2 SBOX 3 [2][13] = 10 SBOX 4 [2][1] = 6 SBOX 5 [2][5] = 13 SBOX 6 [2][1] = 14 SBOX 7 [3][7] = 7 SBOX 8 [2][12] = 15	Runda 11 SBOX 1 [1][11] = 11 SBOX 2 [2][12] = 9 SBOX 3 [0][7] = 5 SBOX 4 [2][4] = 12 SBOX 5 [1][7] = 1 SBOX 6 [2][15] = 6 SBOX 7 [2][6] = 7 SBOX 8 [1][6] = 7
Runda 12 SBOX 1 [0][15] = 7 SBOX 2 [3][3] = 1 SBOX 3 [3][14] = 2 SBOX 4 [1][15] = 9 SBOX 5 [2][9] = 9 SBOX 6 [2][1] = 14 SBOX 7 [3][7] = 7 SBOX 8 [3][14] = 6	Runda 12 SBOX 1 [2][9] = 12 SBOX 2 [3][6] = 4 SBOX 3 [0][9] = 13 SBOX 4 [3][3] = 6 SBOX 5 [2][8] = 15 SBOX 6 [0][7] = 8 SBOX 7 [3][1] = 11 SBOX 8 [3][11] = 0
Runda 13 SBOX 1 [0][9] = 10 SBOX 2 [3][6] = 4 SBOX 3 [0][14] = 2 SBOX 4 [1][3] = 5 SBOX 5 [2][9] = 9 SBOX 6 [2][7] = 3 SBOX 7 [2][1] = 4 SBOX 8 [3][2] = 14	Runda 13 SBOX 1 [3][8] = 5 SBOX 2 [1][11] = 10 SBOX 3 [2][11] = 12 SBOX 4 [3][1] = 15 SBOX 5 [3][14] = 5 SBOX 6 [1][10] = 13 SBOX 7 [0][15] = 1 SBOX 8 [3][7] = 13
Runda 14 SBOX 1 [1][5] = 2 SBOX 2 [2][10] = 12 SBOX 3 [1][5] = 4 SBOX 4 [3][13] = 7 SBOX 5 [2][10] = 12 SBOX 6 [0][3] = 15 SBOX 7 [3][4] = 1 SBOX 8 [0][10] = 3	Runda 14 SBOX 1 [3][6] = 1 SBOX 2 [1][12] = 6 SBOX 3 [1][13] = 11 SBOX 4 [2][14] = 8 SBOX 5 [1][3] = 12 SBOX 6 [2][14] = 11 SBOX 7 [0][7] = 13 SBOX 8 [2][1] = 11
Runda 15 SBOX 1 [0][0] = 14 SBOX 2 [1][5] = 2 SBOX 3 [3][8] = 4 SBOX 4 [1][12] = 1 SBOX 5 [1][10] = 15 SBOX 6 [0][13] = 7 SBOX 7 [3][0] = 6 SBOX 8 [0][10] = 3	Runda 15 SBOX 1 [0][13] = 9 SBOX 2 [3][4] = 3 SBOX 3 [0][8] = 1 SBOX 4 [1][7] = 3 SBOX 5 [2][12] = 6 SBOX 6 [1][3] = 2 SBOX 7 [2][10] = 6 SBOX 8 [1][0] = 1

Runda 16 SBOX 1 [1][11] = 11 SBOX 2 [3][11] = 12 SBOX 3 [3][14] = 2 SBOX 4 [0][10] = 8 SBOX 5 [1][0] = 14 SBOX 6 [1][13] = 11 SBOX 7 [2][12] = 0 SBOX 8 [1][2] = 13	Runda 16 SBOX 1 [3][10] = 3 SBOX 2 [0][8] = 9 SBOX 3 [1][2] = 0 SBOX 4 [0][10] = 8 SBOX 5 [0][7] = 6 SBOX 6 [3][2] = 2 SBOX 7 [0][15] = 1 SBOX 8 [3][3] = 7

Само во првата рунда нема промени во влезот во S-box-овите, потоа во втората рунда има промена само во два s-box-а, а потоа промени има во сите s-box-ови. Оттаму може да заклучиме дека сите S-box-ови се засегнати од промената на едниот бит во пораката, кога извршуваме 16 рунди од алгоритмот.

Задача 2: Кои се слабите клучеви во DES алгоритмот и да се покаже со пример зошто се слаби клучеви.

Дефинирани се 4 слаби клучеви во DES алгоритмот и тоа (хексадецимален запис):

- 0xFFFFFFFFFFFFFFFF
- 0x0000000000000000,
- 0xFFFFFFFF00000000,
- 0x00000000FFFFFFFF

Овие клучеви се наречени слаби клучеви бидејќи кога се формира keystream-от се врши пермутација на битовите и се добиваат 2 половини од по 28 бита, кои што подоцна се ротираат и се шифтираат. Доколку во тие две половини се наоѓаат само нули или само единици, тоа е бескорисно, бидејќи нема да дојде до промена во клучот, тој ќе биде постојано истиот. Конкретно, на пример, шифрираме било која порака со клуч 000.000. Клучот ќе биде 0000.0000 во сите 16 рунди на алгоритмот. Тоа може најдобро да се согледа тука:

```
//zadaca 2
            DES.NUM_OF_ROUNDS=16;
            long M=0xFFFABCD;
            long K=0;
            System.out.println(Long.toBinaryString(des.encrypt(M, K)));
```

Излез: (бр на рунда, шифрирана порака после рундата, клуч за време на рундата)

```
Runda1
11100000011000001111000001110000101011100111011110000011010010 1
Kluc: 0
Runda2
10101110011101111000001101001011111000000101100001111010011000
Kluc: 0
Runda3
11110000001011000011110100110001110010001111011100101010011011 1
Kluc: 0
Runda4
11100100011110111001010100110111100000110100011010010111111011 11
Kluc: 0
Runda5
```

```
1000001101000110100101111110111110010110110101010001001000100101
Kluc: 0
Runda6
1001011011010101000100100010010110011101001010011101100110111
Kluc: 0
Runda7
1001110100101001110110011011110100100101000110100001010100010
Kluc: 0
Runda8
1010010010100011010000101010001011011000111011001111000010001110
Kluc: 0
Runda9
1101100011101100111100001000111010010011011101010001111110101101
Kluc: 0
Runda10
1001001101110101000111110101101110110011101000011000001000101O
Kluc: 0
Runda11
1101100111010000110000010001010101111000000100100000010001110l0
Kluc: 0
Runda12
1011110000001001000000100011101011100100001010100101111100010O0
Kluc: 0
Runda13
1110010000101010010111110001000110011111001111010000100110101l
Kluc: 0
Runda14
1100111110011110100001001101011101011010111111100111100011111OO
Kluc: 0
Runda15
1010110101111111001111000111110010000110100100011111110111011l0
Kluc: 0
Runda16
1000011010010001111111011101110111101010111001000101110110010O1
Kluc: 0
```

Јасно се гледа дека клучот е цело време 0, односно бинарно сите нули.