现代密码学作业——第四讲

第三节

1、

$$\begin{aligned}
&b_3x^3+b_2x^2+b_1x+b_0\\
&=(a_3x^3+a_2x^2+a_1x+a_0)(03x^3+01x^2+01x+02)\\
&=(a_3*03)x^6+(a_3*01+a_2*03)x^5+(a_3*01+a_2*01+a_1*03)x^4\\
&\quad+(a_3*02+a_2*01+a_1*01+a_0*03)x^3+(a_2*02+a_1*01+a_0*01)x^2\\
&\quad+(a_1*02+a_0*01)x+a_0*02\\
&=(a_3*02+a_2*01+a_1*01+a_0*03)x^3+(a_3*03+a_2*02+a_1*01+a_0*01)x^2\\
&\quad+(a_3*01+a_2*03+a_1*02+a_0*01)x+(a_3*01+a_2*01+a_1*03+a_0*02)
\end{aligned}$$

上述公式表达的即为题中所示的矩阵乘法。

2、0x87=1000 0111 即 $a(x)=x^7+x^2+x+1$

0x03=0000 0011

$A^{(0)}=a(x) \bmod m(x)= x^7+x^2+x+1$

$A^{(1)}=x*A^{(0)} \bmod m(x)=x^4+x^2+1$

$a(x)*b(x)=A^{(0)}+A^{(1)}=x^7+x^4+x$

那么，0x87*0x03=1001 0010=0x92

3、

5 密钥及密钥参量

加密密钥长度为128比特，表示为$MK=(MK_0,MK_1,MK_2,MK_3)$，其中$MK_i(i=0,1,2,3)$为字。

轮密钥表示为$(rk_0,rk_1,\cdots,rk_{31})$，其中$rk_i(i=0,\cdots,31)$为32比特字。轮密钥由加密密钥生成。

$FK=(FK_0,FK_1,FK_2,FK_3)$为系统参数，$CK=(CK_0,CK_1,\cdots,CK_{31})$为固定参数，用于密钥扩展算法，其中$FK_i(i=0,\cdots,3)$、$CK_i(i=0,\cdots,31)$为字。

6 轮函数 *F*

6.1 轮函数结构

设输入为$(X_0,X_1,X_2,X_3)\in(Z_2^{32})^4$，轮密钥为$rk\in Z_2^{32}$，则轮函数$F$为：

$$F(X_0,X_1,X_2,X_3,rk)=X_0\oplus T(X_1\oplus X_2\oplus X_3\oplus rk)$$

6.2 合成置换 *T*

$T:Z_2^{32}\rightarrow Z_2^{32}$是一个可逆变换，由非线性变换$\tau$和线性变换$L$复合而成，即$T(.)=L(\tau(.))$。

(1) 非线性变换τ

τ由4个并行的S盒构成。

设输入为$A=(a_0,a_1,a_2,a_3)\in(Z_2^8)^4$，输出为$B=(b_0,b_1,b_2,b_3)\in(Z_2^8)^4$，则

$(b_0,b_1,b_2,b_3)=\tau(A)=(\mathrm{Sbox}(a_0),\mathrm{Sbox}(a_1),\mathrm{Sbox}(a_2),\mathrm{Sbox}(a_3))$

其中，Sbox数据如下：

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	D6	90	E9	FE	CC	E1	3D	B7	16	B6	14	C2	28	FB	2C	05
1	2B	67	9A	76	2A	BE	04	C3	AA	44	13	26	49	86	06	99
2	9C	42	50	F4	91	EF	98	7A	33	54	0B	43	ED	CF	AC	62
3	E4	B3	1C	A9	C9	08	E8	95	80	DF	94	FA	75	8F	3F	A6
4	47	07	A7	FC	F3	73	17	BA	83	59	3C	19	E6	85	4F	A8
5	68	6B	81	B2	71	64	DA	8B	F8	EB	0F	4B	70	56	9D	35
6	1E	24	0E	5E	63	58	D1	A2	25	22	7C	3B	01	21	78	87
7	D4	00	46	57	9F	D3	27	52	4C	36	02	E7	A0	C4	C8	9E
8	EA	BF	8A	D2	40	C7	38	B5	A3	F7	F2	CE	F9	61	15	A1
9	E0	AE	5D	A4	9B	34	1A	55	AD	93	32	30	F5	8C	B1	E3
A	1D	F6	E2	2E	82	66	CA	60	C0	29	23	AB	0D	53	4E	6F
B	D5	DB	37	45	DE	FD	8E	2F	03	FF	6A	72	6D	6C	5B	51
C	8D	1B	AF	92	BB	DD	BC	7F	11	D9	5C	41	1F	10	5A	D8
D	0A	C1	31	88	A5	CD	7B	BD	2D	74	D0	12	B8	E5	B4	B0
E	89	69	97	4A	0C	96	77	7E	65	B9	F1	09	C5	6E	C6	84
F	18	F0	7D	EC	3A	DC	4D	20	79	EE	5F	3E	D7	CB	39	48

注：输入'EF'，则经S盒后的值为表中第E行和第F列的值，Sbox(EF)=84。

(2) 线性变换 L

非线性变换 τ 的输出是线性变换 L 的输入。设输入为 $B\in Z_2^{32}$，输出为 $C\in Z_2^{32}$，则：

$C=L(B)=B\oplus(B<<<2)\oplus(B<<<10)\oplus(B<<<18)\oplus(B<<<24)$

7 算法描述

7.1 加密算法

本加密算法由 32 次迭代运算和 1 次反序变换 R 组成。

设明文输入为 $(X_0,X_1,X_2,X_3)\in(Z_2^{32})^4$，密文输出为 $(Y_0,Y_1,Y_2,Y_3)\in(Z_2^{32})^4$，轮密钥为 $rk_i\in Z_2^{32}$，$i=0,1,2,\cdots,31$。加密算法的运算过程如下：

(1)32 次迭代运算：$X_{i+4}=F(X_i,X_{i+1},X_{i+2},X_{i+3},rk_i)$，$i=0,1,\cdots,31$；

(2)反序变换：$(Y_0,Y_1,Y_2,Y_3)=R(X_{32},X_{33},X_{34},X_{35})=(X_{35},X_{34},X_{33},X_{32})$。

7.2 解密算法

本算法的解密变换与加密变换结构相同，不同的仅是轮密钥的使用顺序。解密时，使用轮密钥序 $(rk_{31},rk_{30},\cdots,rk_0)$。

7.3 密钥扩展算法

本算法轮密钥由加密密钥通过密钥扩展算法生成。

加密密钥 $MK=(MK_0,MK_1,MK_2,MK_3)\in(Z_2^{32})^4$，轮密钥生成方法为：

$(K_0,K_1,K_2,K_3)=(MK_0\oplus FK_0,MK_1\oplus FK_1,MK_2\oplus FK_2,MK_3\oplus FK_3)$，

$rk_i=K_{i+4}=K_i\oplus T'(K_{i+1}\oplus K_{i+2}\oplus K_{i+3}\oplus CK_i)$，$i=0,1,\cdots,31$。

其中：

(1) T'是将 6.2 中合成置换 T 的线性变换 L 替换为 L'：

$L'(B)=B\oplus(B<<<13)\oplus(B<<<23)$；

(2) 系统参数 FK 的取值为：

$FK_0=(A3B1BAC6)$，$FK_1=(56AA3350)$，$FK_2=(677D9197)$，$FK_3=(B27022DC)$；

(3)固定参数 CK 取值方法为：

设 $ck_{i,j}$ 为 CK_i 的第 j 字节($i=0,1,\cdots,31$；$j=0,1,2,3$)，即 $CK_i=(ck_{i,0},ck_{i,1},ck_{i,2},ck_{i,3})\in(Z_2^8)^4$，则 $ck_{i,j}=(4i+j)\times7\ (\mathrm{mod}\ 256)$。

固定参数 $CK_i(i=0,1,\cdots,31)$具体值为：

00070E15， 1C232A31， 383F464D， 545B6269，
70777E85， 8C939AA1， A8AFB6BD，C4CBD2D9，
E0E7EEF5，FC030A11， 181F262D， 343B4249，
50575E65， 6C737A81， 888F969D， A4ABB2B9，
C0C7CED5，DCE3EAF1，F8FF060D， 141B2229，
30373E45， 4C535A61， 686F767D， 848B9299，
A0A7AEB5，BCC3CAD1，D8DFE6ED，F4FB0209，
10171E25， 2C333A41， 484F565D， 646B7279。

4、

```python
# S 盒乘法逆元及仿射实现
# 求最高幂次数
def Nonzero_MSB(value):
    v2str = '{:09b}'.format(value)
    for i in range(9):
        if int(v2str[i]):
            return 9 - i

```

```
# 模 2 除法：m 为被除数。b 为除数，q 为商，r 为余数
def Mode2_div(fx, gx):
    n = Nonzero_MSB(fx)
    m = Nonzero_MSB(gx)
    if n < m:
        return [0, fx]
    deg = n - m
    fx = fx ^ (gx << deg)
    [q, r] = Mode2_div(fx, gx)
    return [(1 << deg) | q, r]

# 多项式乘法
# v3 = v1 - q3 * v2
def Calculate(v1, q3, v2):
    value = 0
    for i in range(32):
        if (q3 & (1 << i)):
            value = value ^ (v2 << i)
    return v1 ^ value

# 欧几里得算法
def poly_gcd(r1, r2, v1=1, v2=0, w1=0, w2=1):
    if r2 == 0 or r2 == 1:
        return w2
    q3, r3 = Mode2_div(r1, r2)  # q3(x)=r1(x)|r2(x), r2(x)=r1(x) mod r2(x)
    v3 = Calculate(v1, q3, v2)  # v3 = v1 - q3 * v2
    w3 = Calculate(w1, q3, w2)  # w3 = w1 - q3 * w2
    return poly_gcd(r2, r3, v2, v3, w2, w3)

def sym2int(sym):
    power = [sym[i + 2] for i in range(len(sym)) if sym[i] == 'x']
    if '+1' in sym: power.append('0')
    data = 0
    for p in power:
        data = data | (1 << int(p))
    return data

def int2sym(data):
    int2str = '{:09b}'.format(data)
    sym = ''
    for i in range(9):
        if int(int2str[i]) == 1:
            if 8 - i:
```

```
                sym += '+x^%d' % (8 - i)
            else:
                sym += '+1'
    return sym[1:]

def xor(a, b):
    if a == b:
        return '0'
    else:
        return '1'

def fangshe(a):
    a = a[::-1]
    c = '11000110'
    b = ''
    for i in range(7, -1, -1):
        b += xor(xor(xor(xor(xor(a[i], a[(i + 4) % 8]), a[(i + 5) % 8]), a[(i + 6) % 8]), a[(i + 7) % 8]), c[i])
    return b

if __name__ == '__main__':
    bi = bin(int(input('请输入两位 16 进制数：'), 16))[2:]
    print("16 进制转为为 2 进制为：{}".format(bi))
    xstr = ''
    for i in range(len(bi)):
        if bi[i] == '1':
            if i < len(bi) - 1:
                xstr += "+x^{}".format(len(bi) - i - 1)
            elif i == str(len(bi)):
                xstr += '+1'
    xstr = xstr[1:]
    inverse = int2sym(poly_gcd(283, sym2int(xstr)))
    data = hex(sym2int(inverse))
    print('你输入的多项式 :', xstr)
    print('默认既约多项式 :', int2sym(283))
    print('乘法逆元为 :', inverse)
    print('乘法逆元的 16 进制：', data)
    print('---- 多项式乘法逆元求解完成 ----')
    data = bin(int(data, 16))[2:].zfill(8)
    print('最终结果为：', fangshe(data))
    print('16 进制表示为', hex(int(fangshe(data), 2)))
```

第四节

1、OFB 模式（输出反馈模式）：密码算法的输出会反馈到密码算法的输入当中。其是将“明文分组”和“密码算法的输出”进行 XOR 来产生“密文分组”。优点：传输过程中密文在某位上发生的错误不会影响解密后明文其他位。缺点：失去同步，如果加密端和解密端移位寄存器不同步，那么恢复的明文将是一些无用的杂乱数据，任何使用 OFB 的系统必须有检测失步的机制，并用新的初始向量 IV 填充双方移位寄存器重新获得同步；抗消息流篡改攻击能力不如 CFB。

2、CCM 模式（认证保密模式）：CCM 是 CTR 加密模式和 CMAC 认证算法的混合使用，常用在需要同时加密和认证的领域，比如 WiFi 安全中的 WPE 协议，它就使用了 AES-CCM 模式。其首先使用 CBC-MAC 模式来认证传输帧，然后使用 CTR 模式来加密帧。