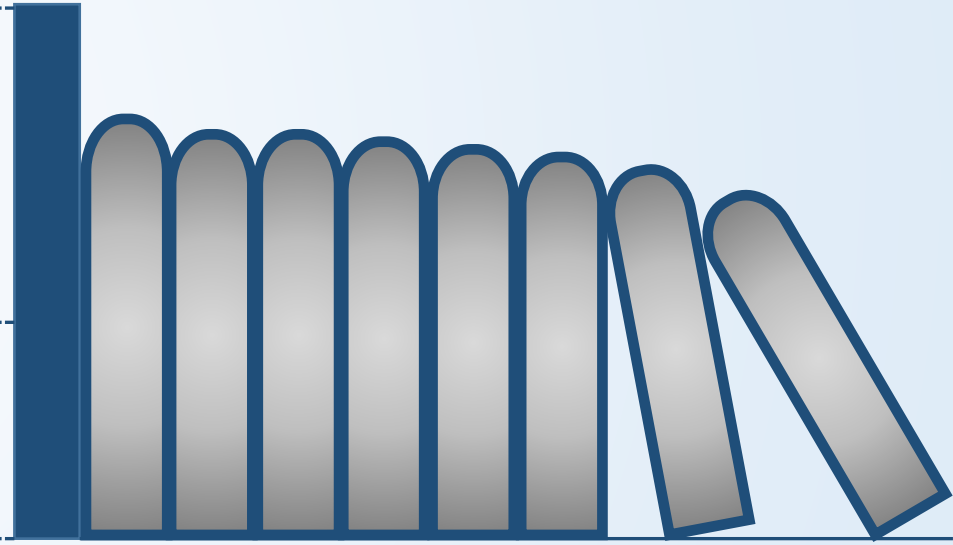
哈爾濱工業大學(深圳)
HARBIN INSTITUTE OF TECHNOLOGY, SHENZHEN

密码学基础实验课程



实验课程安排与考核标准

实验课程共8个学时，4个实验项目，总成绩为30分（30%）。选择综合实践项目可替代4次实验项目，最高也可获得30分。

实验项目

项目编号	实验一	实验二	实验三	实验四
学时数	2	2	2	2
实验项目	AES对称加密算法实验	RSA公钥加密算法实验	Hash长度扩展攻击实验	ElGamal数字签名算法实验
分数数	6	8	6	10

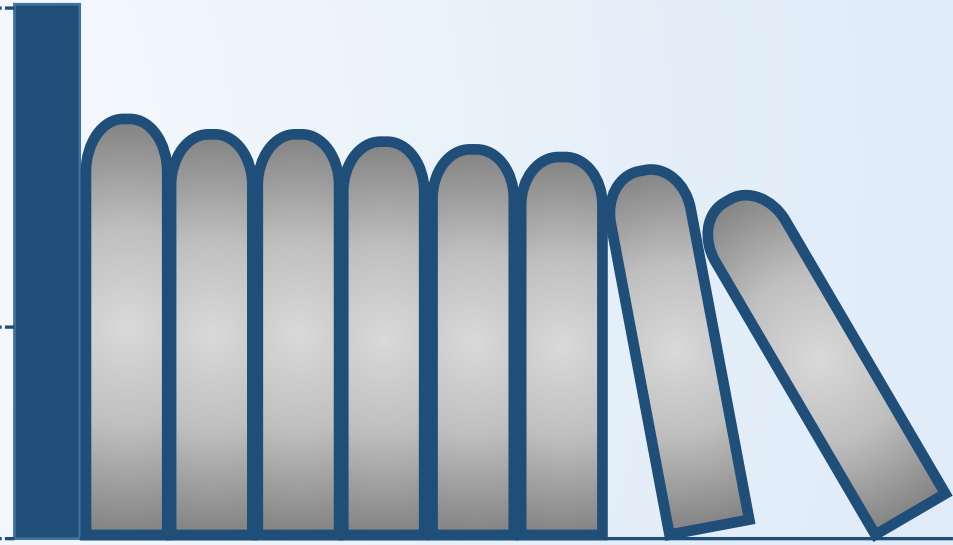
考核方式

- 源代码和结果截图：每次课程均需提交实验程序源代码，以及程序的运行结果截图。
- 实验报告：最后一次课程需提交实验报告。 **禁止抄袭，发现雷同，本次实验双方都是0分。**

实验指导书：https://cryptography.p.cs-lab.top/
课件链接：https://gitee.com/hitsz-cslab/cryptography-labs/tree/master/stupkt

哈爾濱工業大學(深圳)
HARBIN INSTITUTE OF TECHNOLOGY, SHENZHEN

AES密码算法实验



2024年9月24日星期二

实验目的

- 理解分组密码算法的基本思想
- 掌握 AES 算法加密和解密原理
- 掌握AES密钥扩展算法
- 了解不同工作模式的运行方法

实验内容

1、编写程序完成AES128算法的加密和解密过程，请用demo.c中的S盒、逆S盒、轮常量Rcon等信息，但是不要求大家一定用模板，可自行选择编程语言；

2、过程输出10轮K_i的值，要求输出16进制格式；

3、扩展：实现CBC-AES128。（选做）

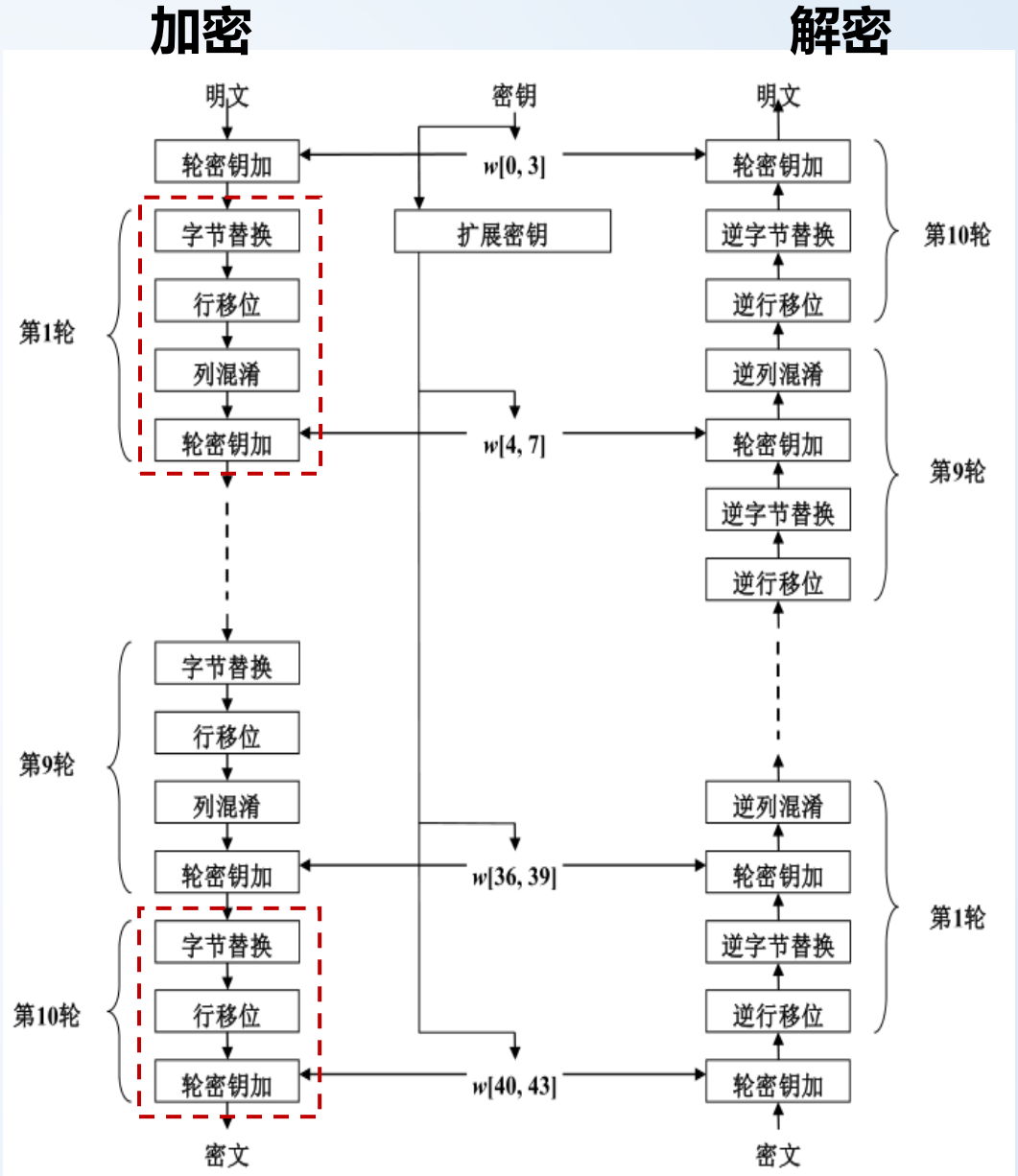
实验指导书：https://cryptography.p.cs-lab.top/
课件链接：https://gitee.com/hitsz-cslab/cryptography-labs/tree/master/stupkt

实验原理

AES算法主要有四种运算：

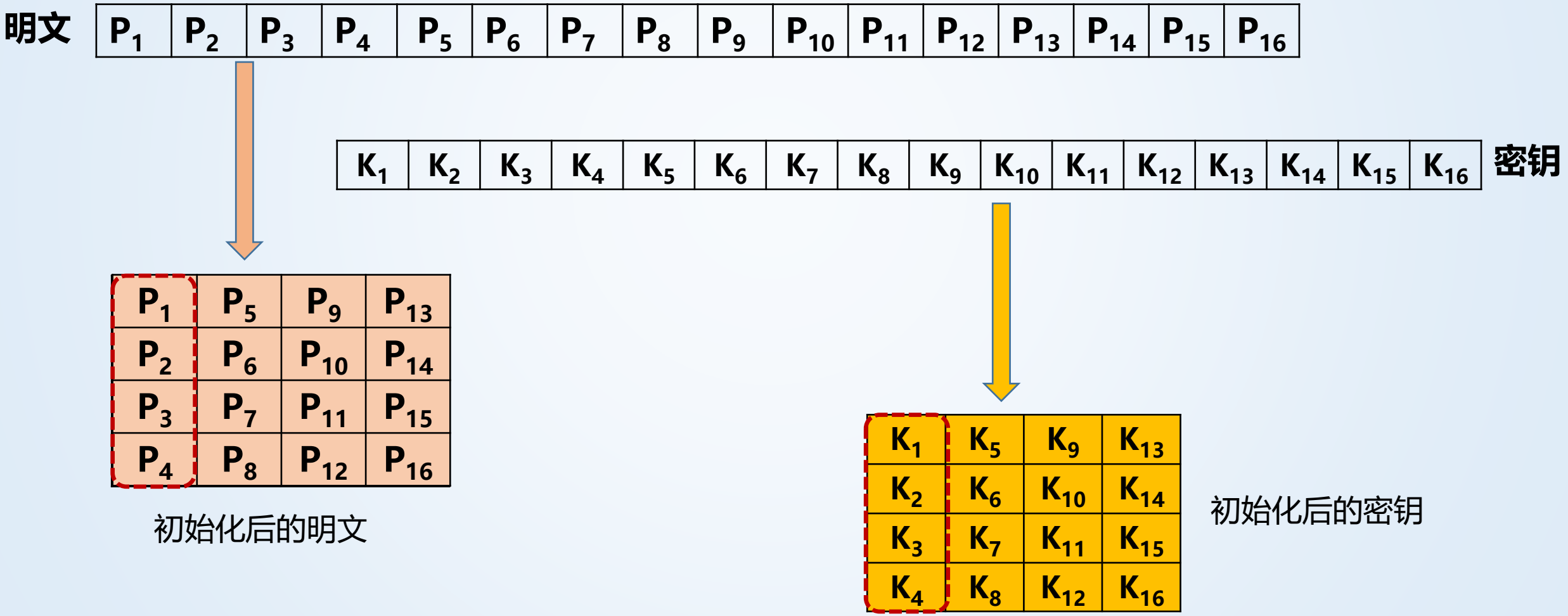
- 字节替换
- 行移位
- 列混淆
- 轮密钥加

- 在轮处理开始前进行了轮密钥加处理
- 最后一轮比前面9轮少了列混淆处理



实验原理

➢ 初始化，AES中的运算都是以字节为单位的，128位的明文可以分为16个字节



实验原理

➢ 字节替换

字节的高4位作为行号，低4位作为列号，查找S盒中对应行列交叉点的元素作为输出。

例如：95对应的输出为2a

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7c	77	7b	f2	6b	6f	c5	30	1	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
A	e0	32	3a	0a	49	6	24	5c	c2	d3	ac	62	91	95	e4	79
B	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
C	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
D	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
E	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
F	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

实验原理

➢ 逆字节替换

字节的高4位作为行号，低4位作为列号，查找逆S盒中对应行列交叉点的元素作为输出。

2a对应的输出为95

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
A	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
B	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
C	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
D	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
E	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
F	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

实验原理

➢ 行移位

- ◆ 每一行按字节循环移位
- ◆ 第1行保持不变，第2行循环左移1个字节，第3行循环左移2个字节，第4行循环左移3个字节
- ◆ 每一列的四个字节被扩散到4个不同的列

a_{00}	a_{01}	a_{02}	a_{03}
a_{10}	a_{11}	a_{12}	a_{13}
a_{20}	a_{21}	a_{22}	a_{23}
a_{30}	a_{31}	a_{32}	a_{33}

输入

行移位

a_{00}	a_{01}	a_{02}	a_{03}
a_{11}	a_{12}	a_{13}	a_{10}
a_{22}	a_{23}	a_{20}	a_{21}
a_{33}	a_{30}	a_{31}	a_{32}

输出

实验原理

➢ 逆行移位

- ◆ 每一行按字节循环移位
- ◆ 第1行保持不变，第2行循环右移1个字节，第3行循环右移2个字节，第4行循环右移3个字节

a_{00}	a_{01}	a_{02}	a_{03}
a_{11}	a_{12}	a_{13}	a_{10}
a_{22}	a_{23}	a_{20}	a_{21}
a_{33}	a_{30}	a_{31}	a_{32}

输入

逆行移位 →

a_{00}	a_{01}	a_{02}	a_{03}
a_{10}	a_{11}	a_{12}	a_{13}
a_{20}	a_{21}	a_{22}	a_{23}
a_{30}	a_{31}	a_{32}	a_{33}

输出

实验原理

➢ 列混淆-左乘一个矩阵

- ◆ 以列为单位，使得输出的每1个字节和输入的4个字节都有关。
- ◆ GF(2^8)上的乘法，模不可约多项式m(x)的乘法运算。

$$\begin{bmatrix} S'_{0,0} & S'_{0,1} & S'_{0,2} & S'_{0,3} \\ S'_{1,0} & S'_{1,1} & S'_{1,2} & S'_{1,3} \\ S'_{2,0} & S'_{2,1} & S'_{2,2} & S'_{2,3} \\ S'_{3,0} & S'_{3,1} & S'_{3,2} & S'_{3,3} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} S_{0,0} & S_{0,1} & S_{0,2} & S_{0,3} \\ S_{1,0} & S_{1,1} & S_{1,2} & S_{1,3} \\ S_{2,0} & S_{2,1} & S_{2,2} & S_{2,3} \\ S_{3,0} & S_{3,1} & S_{3,2} & S_{3,3} \end{bmatrix}$$

输出 **输入**

$$s'_{0,j} = (2 \cdot s_{0,j}) \oplus (3 \cdot s_{1,j}) \oplus s_{2,j} \oplus s_{3,j}$$
$$s'_{1,j} = s_{0,j} \oplus (2 \cdot s_{1,j}) \oplus (3 \cdot s_{2,j}) \oplus s_{3,j}$$
$$s'_{2,j} = s_{0,j} \oplus s_{1,j} \oplus (2 \cdot s_{2,j}) \oplus (3 \cdot s_{2,j})$$
$$s'_{3,j} = (3 \cdot s_{0,j}) \oplus s_{1,j} \oplus s_{2,j} \oplus (2 \cdot s_{2,j})$$

这里·表示GF(2^8)乘法，⊕表示异或操作。

$$3 \cdot s_{1,j} = (02 \oplus 01) \cdot s_{1,j} = (02 \cdot s_{1,j}) \oplus (01 \cdot s_{1,j})$$

实验原理

➢ 逆列混淆

- ◆ 找到逆列混淆的左乘矩阵
- ◆ 逆向列混淆中左乘矩阵与正向列混淆的正向列混淆的左乘矩阵互为逆矩阵

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix}$$

$$\begin{bmatrix} S'_{0,0} S'_{0,1} S'_{0,2} S'_{0,3} \\ S'_{1,0} S'_{1,1} S'_{1,2} S'_{1,3} \\ S'_{2,0} S'_{2,1} S'_{2,2} S'_{2,3} \\ S'_{3,0} S'_{3,1} S'_{3,2} S'_{3,3} \end{bmatrix} = \begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \begin{bmatrix} S_{0,0} S_{0,1} S_{0,2} S_{0,3} \\ S_{1,0} S_{1,1} S_{1,2} S_{1,3} \\ S_{2,0} S_{2,1} S_{2,2} S_{2,3} \\ S_{3,0} S_{3,1} S_{3,2} S_{3,3} \end{bmatrix}$$

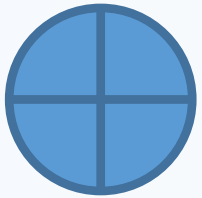
输出 **输入**

实验原理

➢ 轮密钥加

明文矩阵

P_1	P_5	P_9	P_{13}
P_2	P_6	P_{10}	P_{14}
P_3	P_7	P_{11}	P_{15}
P_4	P_8	P_{12}	P_{16}



子密钥矩阵

K_1	K_5	K_9	K_{13}
K_2	K_6	K_{10}	K_{14}
K_3	K_7	K_{11}	K_{15}
K_4	K_8	K_{12}	K_{16}

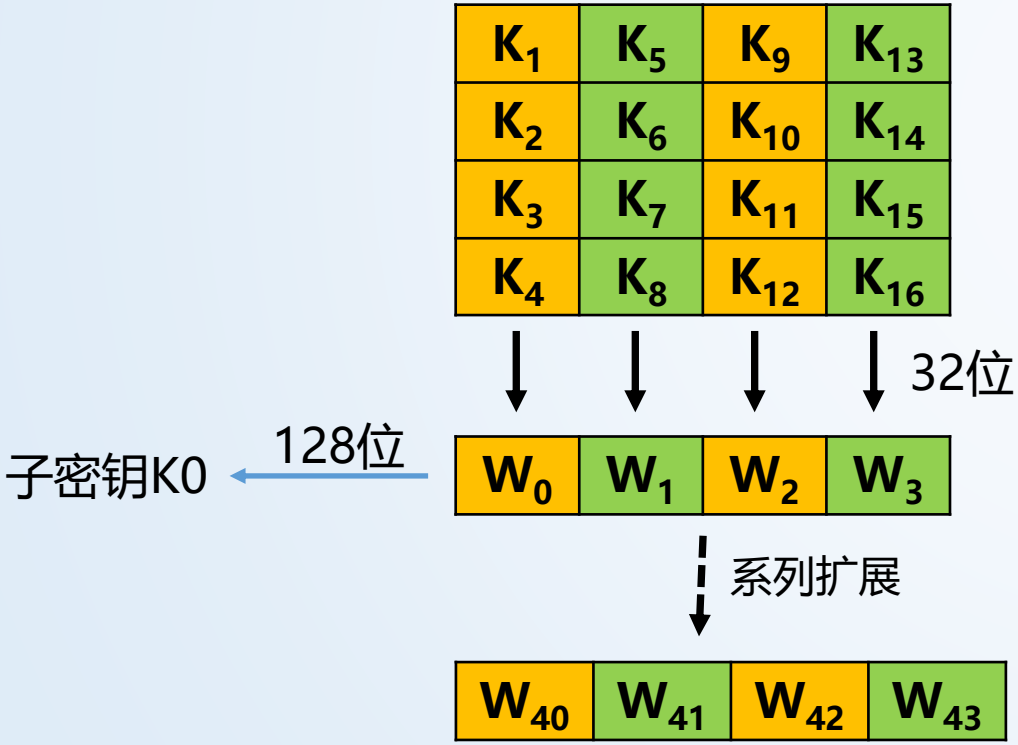
注：将字符转换为ASCii码按字节进行异或

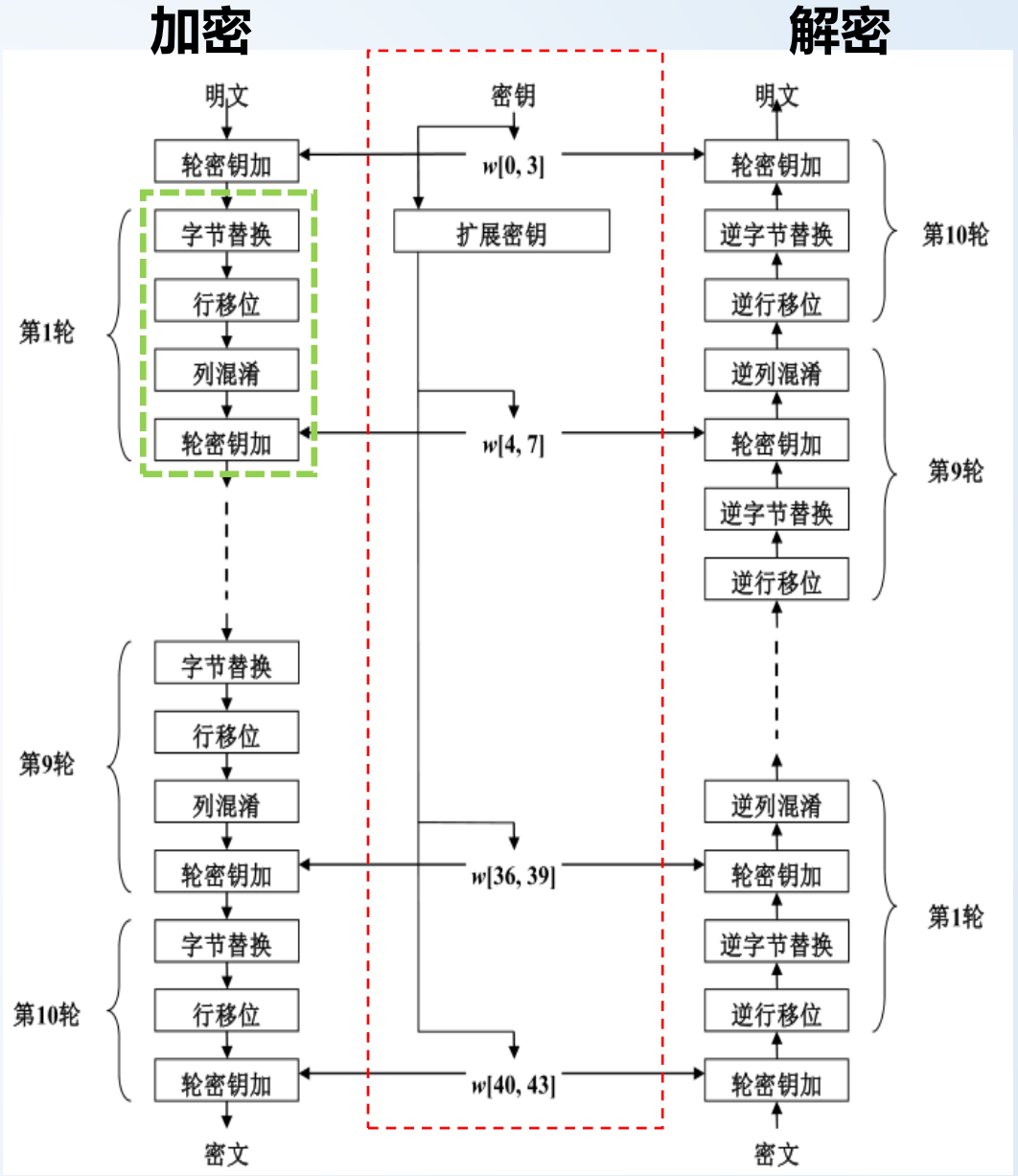
异或的结果再进行异或就是异或的逆

实验原理

➢ 密钥扩展

◆ 由4个字的种子密钥，生成一个44个字的一维线性数组。





实验原理

➢ 密钥扩展

◆ 当i<4时

W[0]=($k_1k_2k_3k_4$)

W[1]=($k_5k_6k_7k_8$)

W[2]=($k_9k_{10}k_{11}k_{12}$)

W[3]=($k_{13}k_{14}k_{15}k_{16}$)

◆ 当i>=4时，其中N_k=4

$$W[i]=\begin{cases} W[i-N_k]\oplus temp & (i \bmod N_k = 0) \\ W[i-N_k]\oplus W[i-1] & (i \bmod N_k \neq 0) \end{cases}$$

temp=SubByte (RotByte (*W*[*i*-1])) ⊕ Rcon[j]

RotByte ( )表示循环左移一个字节；

SubByte( )是S盒的字节代换；

Rcon[j]为轮常数，其中j是轮数。

表 1 轮常量值表

j	1	2	3	4	5
Rcon[j]	01000000	02000000	04000000	08000000	10000000
j	6	7	8	9	10
Rcon[j]	20000000	40000000	80000000	1B000000	36000000

```
/************************************
Nk: 10轮的密钥取4，12轮取6
Nb:  一轮密钥字长度（字位单位），这里取4
Nr:  轮数，这里取10
************************************/
KeyExpansion (byte    Key[4*Nk] , W[Nb*(Nr+1)])
{
    For (i =0; i < Nk; i ++)
        W[i]=( Key[4* i], Key[4* i +1], Key[4* i +2], Key[4* i +3] );
    For (i =Nk; i <Nb*(Nr+1); i ++)
    {
        temp=W[i -1];
        if (i % Nk= =0)
            temp=SubByte (RotByte (temp))^Rcon[i /Nk];
        W[i]=W[i - Nk]^temp;
    }
}
```

实验原理

➢ 密钥扩展

举个例子，种子密钥K= 06 07 08 09 0A 0B 0C 0D 0E 0F 00 01 02 03 04 05

W_0 = 06 07 08 09
W_1 = 0A 0B 0C 0D
W_2 = 0E 0F 00 01
W_3 = 02 03 04 05

$$K_0 = (W_0, W_1, W_2, W_3) = \begin{bmatrix} 06 & 0A & 0E & 02 \\ 07 & 0B & 0F & 03 \\ 08 & 0C & 00 & 04 \\ 09 & 0D & 01 & 05 \end{bmatrix}$$

W_4 = SubByte (RotByte (W_3)) ⊕ Rcon[1] ⊕ W_0

= SubByte(03 04 05 02) ⊕ Rcon[1] ⊕W_0

= (7B F2 6B 77) ⊕(01 00 00 00) ⊕ (06 07 08 09)

= 7C F5 63 7E

W_5 = W_1 ⊕ W_4 = (0A 0B 0C 0D) ⊕ (7C F5 63 7E) = 76 FE 6F 73

W_6 = W_2 ⊕ W_5 = (0E 0F 00 01) ⊕ (76 FE 6F 73) = 78 F1 6F 72

W_7 = W_3 ⊕ W_6 = (02 03 04 05) ⊕ (78 F1 6F 72) = 7A F2 6B 77

$$K_1 = (W_4, W_5, W_6, W_7) = \begin{bmatrix} 7C & 76 & 78 & 7A \\ F5 & FE & F1 & F2 \\ 63 & 6F & 6F & 6B \\ 7E & 73 & 72 & 77 \end{bmatrix}$$

实验步骤

加密

Step1.完成初始化存储函数 void convertToIntArray(char *str, int pa[4][4])，按照列存储；

Step2.完成字节代换函数 void subBytes(int array[4][4])，利用已经给出的函数getNumFromSBox()；

Step3.实现左移函数void leftLoop4int(int array[4], int step)和行移位函数void shiftRows(int array[4][4])；

Step4.实验列混淆函数void mixColumns(int array[4][4])，利用已经给出的函数GFMul(int n, int s)；

Step5.实现轮密钥加函数void addRoundKey(int array[4][4], int round)；

Step6.实现密钥扩展中的T函数int T(int num, int round)；

Step7.实验密钥扩展函数void extendKey(char *key)；

Step8.读懂aes函数为完成deaes函数做准备。

解密

Step1.完成逆字节代换函数 void deSubBytes(int array[4][4])，利用已经给出的函数getNumFromS1Box()；

Step2.实现右移函数void rightLoop4int(int array[4], int step)和行移位函数void deShiftRows(int array[4][4])；

Step3.实验逆列混淆函数void deMixColumns(int array[4][4])，利用已经给出的函数GFMul(int n, int s)；

Step4.完成void deAes(char *c, int clen, char *key)

实验要求

- 提交内容

① 源代码

② 实验结果截图

- 截止时间

下一次实验课前提交至HITsz Grader 作业提交平台，具体截止日期参考平台发布。

- 登录网址：： http://grader.tery.top:8000/#/login
- 推荐浏览器： Chrome
- 初始用户名、密码均为学号，登录后请修改

请同学们开始实验