

公告

昵称: yxwkaifa
园龄: 2年10个月
粉丝: 21
关注: 0
+加关注

< 2017年4月 >

日 一 二 三 四 五 六

26 27 28 29 30 31 1

2 3 4 5 6 7 8

9 10 11 12 13 14 15

16 17 18 19 20 21 22

23 24 25 26 27 28 29

30 1 2 3 4 5 6

搜索

找找看

谷歌搜索

常用链接

我的随笔

我的评论

我的参与

最新评论

我的标签

随笔档案

2016年4月 (322)

2016年3月 (325)

2016年2月 (251)

2016年1月 (387)

2015年12月 (273)

2015年10月 (105)

2015年9月 (274)

2015年8月 (242)

2015年7月 (340)

2015年6月 (229)

2015年5月 (39)

2014年11月 (175)

2014年10月 (302)

2014年9月 (87)

2014年8月 (252)

2014年7月 (416)

2014年6月 (75)

2014年5月 (2)

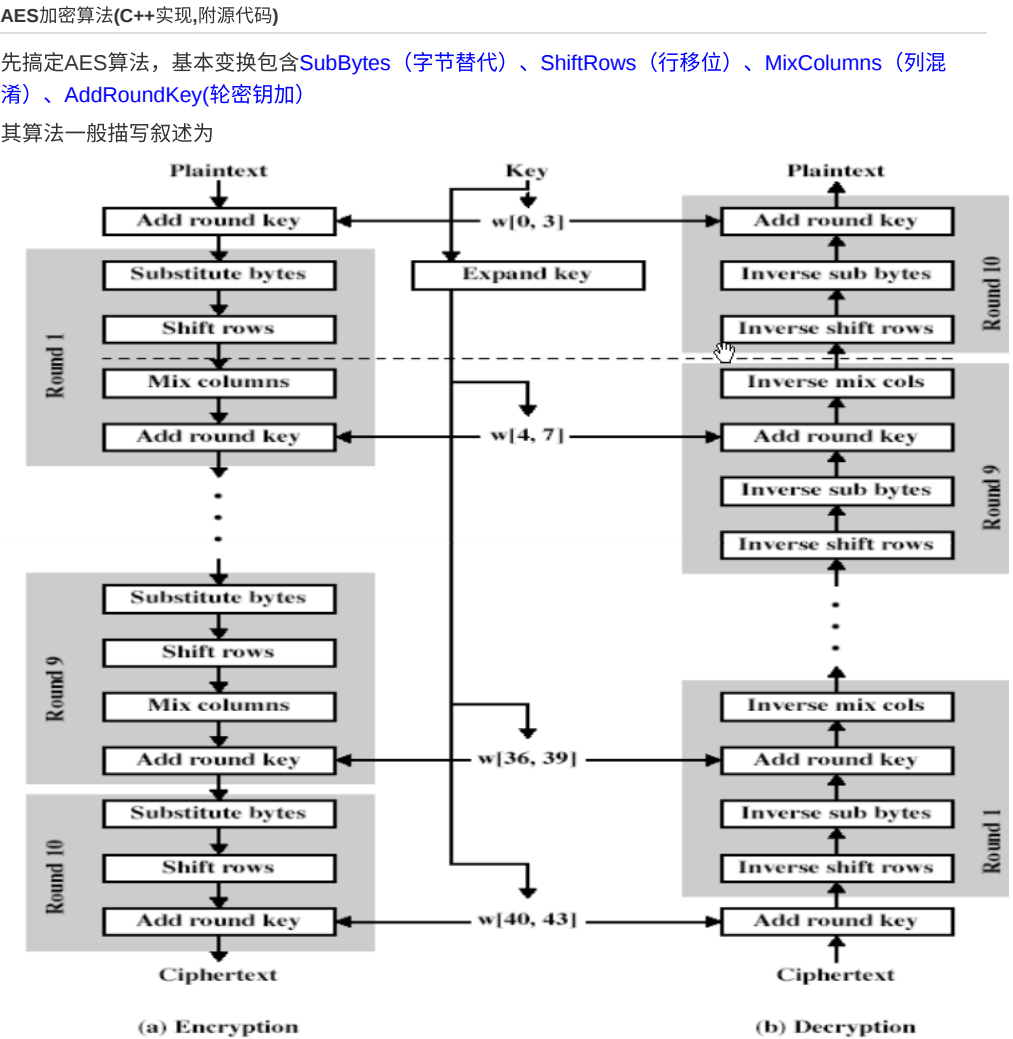
最新评论

1. Re:PCB设计资料: 看到最后才知道是福利
厉害了我的楼主大大
--_start

2. Re:做高通平台安卓驱动感言
你好，看到你的经验想向你多学习，看到后能否
加我qq： 2856763420
--FengShen

3. Re:.net web 开发平台- 表单设计器 一(web版)
在吗，最近正在为这个自定义报表发愁，能分享
一下吗，买也可以的？ 470706214@qq.com
--禅道

4. Re:Java SE学习之printf 日期转换符
我将计算机的语言切换成英语后，运行结果和上
面的表一样了
--羽翼未丰之冲天槊



ByteSubstitution（字节替代）

非线性的字节替代，单独处理每一个字节：

求该字节在有限域GF(2⁸)上的乘法逆，“0”被映射为自身，即对于α∈GF(2⁸），求β∈GF(2⁸），使得α·β=β·α=1mod(x⁸+x⁴+x²+x+1)。

对上一步求得的乘法逆作仿射变换

$$Y_i = X_i + X_{(i+4) \bmod 8} + X_{(i+6) \bmod 8} + X_{(i+7) \bmod 8} + C_i$$

(当中C_i是63₁₀即01100011₂的第i位），用矩阵表示为

5. Re:Android Fragment简单实例

--者旨於陽

阅读排行榜

1. 汇报措辞：你懂得如何向领导汇报吗（审阅、审批、审阅、批示、查阅）？(3881)
2. Microsoft Visual C++ Runtime Library Runtime Error的解决的方法(1847)
3. 自己写一个jquery的拖拽插件(1684)
4. iOS国际化时遇到的错误:read failed: the data couldn't be read because it isn't in the correct format.(1459)
5. 不可错过的手机APP常见8种界面导航样式(1316)

评论排行榜

1. PCB设计资料：看到最后才知道是福利(2)
2. .net web 开发平台- 表单设计器 一(web版)(2)
3. Android Fragment简单实例(1)
4. Java SE学习之printf 日期转换符(1)
5. 做高通平台安卓驱动感言(1)

推荐排行榜

1. Struts2中属性驱动与模型驱动(1)
2. PCB设计资料：看到最后才知道是福利(1)
3. 西门子PLC学习笔记二-（工作记录）(1)
4. linux学习 建立静态库，动态库，写简单的makefile(1)
5. 几种常见模式识别算法整理和总结(1)

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

本来打算把求乘法逆和仿射变换算法敲上去，最后还是放弃了...直接打**置换表**

```

1 unsigned
2 char
3
4 sBox[] =
5 {
6     /*
7         0      1      2      3      4      5      6      7      8      9      a      b      c      d      e      f */
8
9         0x63, 0x7c, 0x77, 0x7b, 0xf2, 0x6b, 0x6f, 0xc5, 0x30, 0x01, 0x67, 0x2b, 0xfe, 0xd7, 0xab, 0x76,
10
11     /*0*/
12
13         0xca, 0x82, 0xc9, 0x7d, 0xfa, 0x59, 0x47, 0xf0, 0xad, 0xd4, 0xa2, 0xaf, 0x9c, 0xa4, 0x72, 0xc0,
14
15     /*1*/
16
17         0xb7, 0xfd, 0x93, 0x26, 0x36, 0x3f, 0xf7, 0xcc, 0x34, 0xa5, 0xe5, 0xf1, 0x71, 0xd8, 0x31, 0x15,
18
19     /*2*/
20
21         0x04, 0xc7, 0x23, 0xc3, 0x18, 0x96, 0x05, 0x9a, 0x07, 0x12, 0x80, 0xe2, 0xeb, 0x27, 0xb2, 0x75,
22
23     /*3*/
24
25         0x09, 0x83, 0x2c, 0x1a, 0x1b, 0x6e, 0x5a, 0xa0, 0x52, 0x3b, 0xd6, 0xb3, 0x29, 0xe3, 0x2f, 0x84,
26
27     /*4*/
28
29         0x53, 0xd1, 0x00, 0xed, 0x20, 0xfc, 0xb1, 0x5b, 0x6a, 0xcb, 0xbe, 0x39, 0x4a, 0x4c, 0x58, 0xcf,
30
31     /*5*/
32
33         0xd0, 0xef, 0xaa, 0xfb, 0x43, 0x4d, 0x33, 0x85, 0x45, 0xf9, 0x02, 0x7f, 0x50, 0x3c, 0x9f, 0xa8,
34
35     /*6*/
36
37         0x51, 0xa3, 0x40, 0x8f, 0x92, 0x9d, 0x38, 0xf5, 0xbc, 0xb6, 0xda, 0x21, 0x10, 0xff, 0xf3, 0xd2,
38
39     /*7*/
40
41         0xcd, 0x0c, 0x13, 0xec, 0x5f, 0x97, 0x44, 0x17, 0xc4, 0xa7, 0x7e, 0x3d, 0x64, 0x5d, 0x19, 0x73,
42
43     /*8*/
44
45         0x60, 0x81, 0x4f, 0xdc, 0x22, 0x2a, 0x90, 0x88, 0x46, 0xee, 0xb8, 0x14, 0xde, 0x5e, 0x0b, 0xdb,
46
47     /*9*/
48
49         0xe0, 0x32, 0x3a, 0x0a, 0x49, 0x06, 0x24, 0x5c, 0xc2, 0xd3, 0xac, 0x62, 0x91, 0x95, 0xe4, 0x79,
50
51     /*a*/
52
53         0xe7, 0xc8, 0x37, 0x6d, 0x8d, 0xd5, 0x4e, 0xa9, 0x6c, 0x56, 0xf4, 0xea, 0x65, 0x7a, 0xae, 0x08,
54
55     /*b*/
56
57         0xba, 0x78, 0x25, 0x2e, 0x1c, 0xa6, 0xb4, 0xc6, 0xe8, 0xdd, 0x74, 0x1f, 0x4b, 0xbd, 0x8b, 0x8a,
58
59     /*c*/
60
61         0x70, 0x3e, 0xb5, 0x66, 0x48, 0x03, 0xf6, 0x0e, 0x61, 0x35, 0x57, 0xb9, 0x86, 0xc1, 0x1d, 0x9e,
62
63     /*d*/
64
65         0x05, 0x0b, 0x94, 0xe6, 0x65, 0x4a, 0xa0, 0x8c, 0xf8, 0xef, 0x8b, 0xb9, 0x64, 0x7d, 0x11, 0x0a,
66
67     /*e*/
68
69         0x27, 0xb3, 0x31, 0x15, 0x4c, 0x02, 0x77, 0x1a, 0xd9, 0x54, 0x0d, 0x6f, 0x55, 0x21, 0x63, 0x68,
70
71     /*f*/
72
73         0x13, 0x0e, 0x4a, 0x08, 0x27, 0xc1, 0x23, 0x14, 0x40, 0x7f, 0x50, 0x3c, 0x9f, 0xa8, 0xd0, 0xef,
74
75     /*g*/
76
77         0x09, 0x83, 0x2c, 0x1a, 0x1b, 0x6e, 0x5a, 0xa0, 0x52, 0x3b, 0xd6, 0xb3, 0x29, 0xe3, 0x2f, 0x84,
78
79     /*h*/
80
81         0x53, 0xd1, 0x00, 0xed, 0x20, 0xfc, 0xb1, 0x5b, 0x6a, 0xcb, 0xbe, 0x39, 0x4a, 0x4c, 0x58, 0xcf,
82
83     /*i*/
84
85         0xd0, 0xef, 0xaa, 0xfb, 0x43, 0x4d, 0x33, 0x85, 0x45, 0xf9, 0x02, 0x7f, 0x50, 0x3c, 0x9f, 0xa8,
86
87     /*j*/
88
89         0x51, 0xa3, 0x40, 0x8f, 0x92, 0x9d, 0x38, 0xf5, 0xbc, 0xb6, 0xda, 0x21, 0x10, 0xff, 0xf3, 0xd2,
90
91     /*k*/
92
93         0xcd, 0x0c, 0x13, 0xec, 0x5f, 0x97, 0x44, 0x17, 0xc4, 0xa7, 0x7e, 0x3d, 0x64, 0x5d, 0x19, 0x73,
94
95     /*l*/
96
97         0x60, 0x81, 0x4f, 0xdc, 0x22, 0x2a, 0x90, 0x88, 0x46, 0xee, 0xb8, 0x14, 0xde, 0x5e, 0x0b, 0xdb,
98
99     /*m*/
100
101         0xe0, 0x32, 0x3a, 0x0a, 0x49, 0x06, 0x24, 0x5c, 0xc2, 0xd3, 0xac, 0x62, 0x91, 0x95, 0xe4, 0x79,
102
103     /*n*/
104
105         0xe7, 0xc8, 0x37, 0x6d, 0x8d, 0xd5, 0x4e, 0xa9, 0x6c, 0x56, 0xf4, 0xea, 0x65, 0x7a, 0xae, 0x08,
106
107     /*o*/
108
109         0xba, 0x78, 0x25, 0x2e, 0x1c, 0xa6, 0xb4, 0xc6, 0xe8, 0xdd, 0x74, 0x1f, 0x4b, 0xbd, 0x8b, 0x8a,
110
111     /*p*/
112
113         0x70, 0x3e, 0xb5, 0x66, 0x48, 0x03, 0xf6, 0x0e, 0x61, 0x35, 0x57, 0xb9, 0x86, 0xc1, 0x1d, 0x9e,
114
115     /*q*/
116
117         0x05, 0x0b, 0x94, 0xe6, 0x65, 0x4a, 0xa0, 0x8c, 0xf8, 0xef, 0x8b, 0xb9, 0x64, 0x7d, 0x11, 0x0a,
118
119     /*r*/
120
121         0x27, 0xb3, 0x31, 0x15, 0x4c, 0x02, 0x77, 0x1a, 0xd9, 0x54, 0x0d, 0x6f, 0x55, 0x21, 0x63, 0x68,
122
123     /*s*/
124
125         0x13, 0x0e, 0x4a, 0x08, 0x27, 0xc1, 0x23, 0x14, 0x40, 0x7f, 0x50, 0x3c, 0x9f, 0xa8, 0xd0, 0xef,
126
127     /*t*/
128
129         0x09, 0x83, 0x2c, 0x1a, 0x1b, 0x6e, 0x5a, 0xa0, 0x52, 0x3b, 0xd6, 0xb3, 0x29, 0xe3, 0x2f, 0x84,
130
131     /*u*/
132
133         0x53, 0xd1, 0x00, 0xed, 0x20, 0xfc, 0xb1, 0x5b, 0x6a, 0xcb, 0xbe, 0x39, 0x4a, 0x4c, 0x58, 0xcf,
134
135     /*v*/
136
137         0xd0, 0xef, 0xaa, 0xfb, 0x43, 0x4d, 0x33, 0x85, 0x45, 0xf9, 0x02, 0x7f, 0x50, 0x3c, 0x9f, 0xa8,
138
139     /*w*/
140
141         0x51, 0xa3, 0x40, 0x8f, 0x92, 0x9d, 0x38, 0xf5, 0xbc, 0xb6, 0xda, 0x21, 0x10, 0xff, 0xf3, 0xd2,
142
143     /*x*/
144
145         0xcd, 0x0c, 0x13, 0xec, 0x5f, 0x97, 0x44, 0x17, 0xc4, 0xa7, 0x7e, 0x3d, 0x64, 0x5d, 0x19, 0x73,
146
147     /*y*/
148
149         0x60, 0x81, 0x4f, 0xdc, 0x22, 0x2a, 0x90, 0x88, 0x46, 0xee, 0xb8, 0x14, 0xde, 0x5e, 0x0b, 0xdb,
150
151     /*z*/
152
153         0xe0, 0x32, 0x3a, 0x0a, 0x49, 0x06, 0x24, 0x5c, 0xc2, 0xd3, 0xac, 0x62, 0x91, 0x95, 0xe4, 0x79,
154
155     /*aa*/
156
157         0xe7, 0xc8, 0x37, 0x6d, 0x8d, 0xd5, 0x4
```

```
/*d*/

    0xe1, 0xf8, 0x98, 0x11, 0x69, 0xd9, 0x8e, 0x94, 0x9b, 0x1e, 0x87, 0xe9, 0xce, 0x55, 0x28, 0xdf,
/*e*/

    0x8c, 0xa1, 0x89, 0x0d, 0xbf, 0xe6, 0x42, 0x68, 0x41, 0x99, 0x2d, 0x0f, 0xb0, 0x54, 0xbb, 0x16
/*f*/

};
```

以下是逆置换表，解密时使用

```
1 unsigned
2 char
3
4 invsBox[256] =
5
6 {
7     /*
8     0    1    2    3    4    5    6    7    8    9    a    b    c    d    e    f */
9
10    0x52, 0x09, 0x6a, 0xd5, 0x30, 0x36, 0xa5, 0x38, 0xbf, 0x40, 0xa3, 0x9e, 0x81, 0xf3, 0xd7, 0xfb,
11    /*0*/
12
13    0x7c, 0xe3, 0x39, 0x82, 0x9b, 0x2f, 0xff, 0x87, 0x34, 0x8e, 0x43, 0x44, 0xc4, 0xde, 0xe9, 0xcb,
14    /*1*/
15
16    0x54, 0x7b, 0x94, 0x32, 0xa6, 0xc2, 0x23, 0x3d, 0xee, 0x4c, 0x95, 0x0b, 0x42, 0xfa, 0xc3, 0x4e,
17    /*2*/
18
19    0x08, 0x2e, 0xa1, 0x66, 0x28, 0xd9, 0x24, 0xb2, 0x76, 0x5b, 0xa2, 0x49, 0x6d, 0x8b, 0xd1, 0x25,
20    /*3*/
21
22    0x72, 0xf8, 0xf6, 0x64, 0x86, 0x68, 0x98, 0x16, 0xd4, 0xa4, 0x5c, 0xcc, 0x5d, 0x65, 0xb6, 0x92,
23    /*4*/
24
25    0x6c, 0x70, 0x48, 0x50, 0xfd, 0xed, 0xb9, 0xda, 0x5e, 0x15, 0x46, 0x57, 0xa7, 0x8d, 0x9d, 0x84,
26    /*5*/
27
28    0x90, 0xd8, 0xab, 0x00, 0x8c, 0xbc, 0xd3, 0x0a, 0xf7, 0xe4, 0x58, 0x05, 0xb8, 0xb3, 0x45, 0x06,
29    /*6*/
30
31    0xd0, 0x2c, 0x1e, 0x8f, 0xca, 0x3f, 0x0f, 0x02, 0xc1, 0xaf, 0xbd, 0x03, 0x01, 0x13, 0x8a, 0x6b,
32    /*7*/
33
34    0x3a, 0x91, 0x11, 0x41, 0x4f, 0x67, 0xdc, 0xea, 0x97, 0xf2, 0xcf, 0xce, 0xf0, 0xb4, 0xe6, 0x73,
35    /*8*/
36
37    0x96, 0xac, 0x74, 0x22, 0xe7, 0xad, 0x35, 0x85, 0xe2, 0xf9, 0x37, 0xe8, 0x1c, 0x75, 0xdf, 0x6e,
38    /*9*/
39
40    0x47, 0xf1, 0x1a, 0x71, 0x1d, 0x29, 0xc5, 0x89, 0x6f, 0xb7, 0x62, 0x0e, 0xaa, 0x18, 0xbe, 0x1b,
41    /*a*/
42
43    0xfc, 0x56, 0x3e, 0x4b, 0xc6, 0xd2, 0x79, 0x20, 0x9a, 0xdb, 0xc0, 0xfe, 0x78, 0xcd, 0x5a, 0xf4,
44    /*b*/
45
46    0x1f, 0xdd, 0xa8, 0x33, 0x88, 0x07, 0xc7, 0x31, 0xb1, 0x12, 0x10, 0x59, 0x27, 0x80, 0xec, 0x5f,
47    /*c*/
48
49    0x60, 0x51, 0x7f, 0xa9, 0x19, 0xb5, 0x4a, 0x0d, 0x2d, 0xe5, 0x7a, 0x9f, 0x93, 0xc9, 0x9c, 0xef,
50    /*d*/
```

```
0xa0, 0xe0, 0x3b, 0x4d, 0xae, 0x2a, 0xf5, 0xb0, 0xc8, 0xeb, 0xbb, 0x3c, 0x83, 0x53, 0x99, 0x61,
/*e*/

0x17, 0x2b, 0x04, 0x7e, 0xba, 0x77, 0xd6, 0x26, 0xe1, 0x69, 0x14, 0x63, 0x55, 0x21, 0x0c, 0x7d
/*f*/

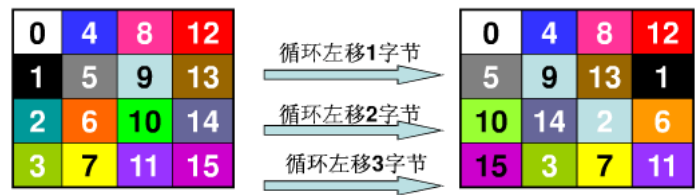
};
```

这里遇到问题了，本来用纯c初始化数组非常正常，封装成类以后发现不能初始化，无论是声明、构造函数都无法初始化，百歌谷度了一通后没有不论什么答案，无奈仅仅能在构造函数中声明一个局部变量数组并初始化，然后用memcpy，（成员变量名为Sbox/InvSbox，局部变量名sBox/invSBox）

```
1 void
2 AES::SubBytes(unsigned char
3 state[][4])
4 {
5     int
6     r,c;
7     for(r=0;
8         r<4; r++)
9     {
10         for(c=0;
11             c<4; c++)
12         {
13             state[r][c]
14             = Sbox[state[r][c]];
15         }
16     }
17 }
```

ShiftRows（行移位变换）

行移位变换完毕基于行的循环位移操作，变换方法：



即行移位变换作用于行上，第0行不变，第1行循环左移1个字节，第2行循环左移2个字节，第3行循环左移3个字节。

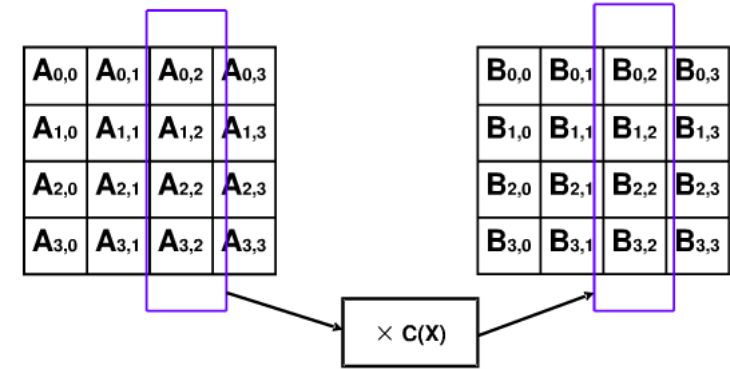
```
1 void
2 AES::ShiftRows(unsigned char
```

```
3 state[][4])
4 {
5     unsigned
6     char
7     t[4];
8     int
9     r,c;
10    for(r=1;
11        r<4; r++)
12    {
13        for(c=0;
14            c<4; c++)
15        {
16            t[c]
17            = state[r][(c+r)%4];
18        }
19        for(c=0;
20            c<4; c++)
21        {
22            state[r][c]
23            = t[c];
24        }
25    }
26 }
```

MixColumns（列混淆变换）

逐列混合，方法：

$$b(x) = (03 \cdot x^3 + 01 \cdot x^2 + 01 \cdot x + 02) \cdot a(x) \bmod (x^4 + 1)$$



矩阵表示形式：

$$\begin{bmatrix} \mathbf{b}_0 \\ \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{02} & \mathbf{03} & \mathbf{01} & \mathbf{01} \\ \mathbf{01} & \mathbf{02} & \mathbf{03} & \mathbf{01} \\ \mathbf{01} & \mathbf{01} & \mathbf{02} & \mathbf{03} \\ \mathbf{03} & \mathbf{01} & \mathbf{01} & \mathbf{02} \end{bmatrix} \begin{bmatrix} \mathbf{a}_0 \\ \mathbf{a}_1 \\ \mathbf{a}_2 \\ \mathbf{a}_3 \end{bmatrix}$$

```

1  void
2  AES::MixColumns(unsigned char
3  state[][4])
4  {
5      unsigned
6      char
7
8      t[4];
9
10     int
11     r, c;
12
13     for(c=0;
14         c< 4; c++)
15     {
16         for(r=0;
17             r<4; r++)
18         {
19             t[r]
20             = state[r][c];
21
22             state[r][c]
23             = FFmul(0x02, t[r])
24               ^
25               FFmul(0x03, t[(r+1)%4])
26               ^
27               FFmul(0x01, t[(r+2)%4])
28               ^
29               FFmul(0x01, t[(r+3)%4]);
30         }
31     }
32 }

```

```
29
30 unsigned
31 char
32 AES::FFmul(unsigned char
33 a, unsigned char
34 b)
35 {
36     unsigned
37     char
38     bw[4];
39     unsigned
40     char
41     res=0;
42     int
43     i;
44     bw[0]
45     = b;
46
47     for(i=1;
48         i<4; i++)
49     {
50         bw[i]
51         = bw[i-1]<<1;
52
53         if(bw[i-1]&0x80)
54         {
55             bw[i]^=0x1b;
56         }
57     }
58
59     for(i=0;
60         i<4; i++)
61     {
62         if((a>>i)&0x01)
63         {
64             res
65             ^= bw[i];
66         }
67     }
68 }
```

```

        return

    res;

}

```

当中FFmul为有限域GF(2⁸)上的乘法，标准算法应该是循环8次（b与a的每一位相乘，结果相加），但这里仅仅用到最低2位，解密时用到的逆列混淆也仅仅用了低4位，所以在这里高4位的运算是多余的，仅仅计算低4位。

AddRoundKey（轮密钥加变换）

简单来说就是逐字节相加，有限域GF(2⁸)上的加法是模2加法，即异或

```

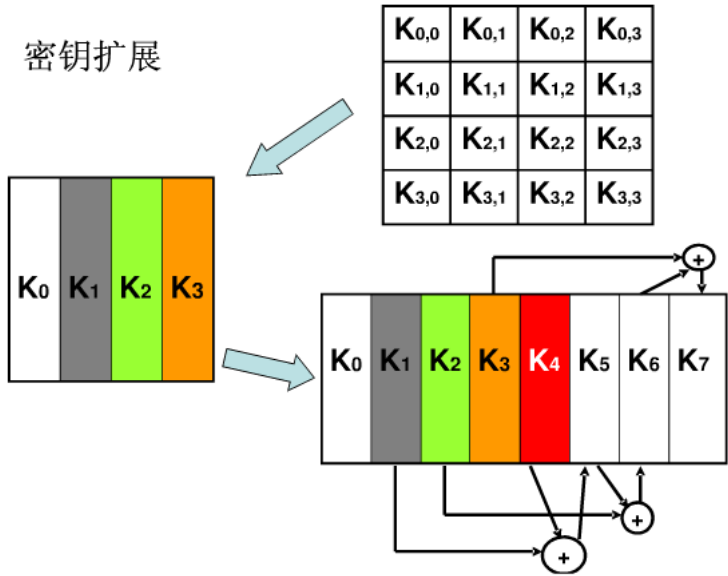
1  void
2  AES::AddRoundKey(unsigned char
3  state[][4], unsigned char
4  k[][4])
5  {
6      int
7      r,c;
8      for(c=0;
9          c<4; c++)
10     {
11         for(r=0;
12             r<4; r++)
13         {
14             state[r][c]
15             ^= k[r][c];
16         }
17     }
18 }

```

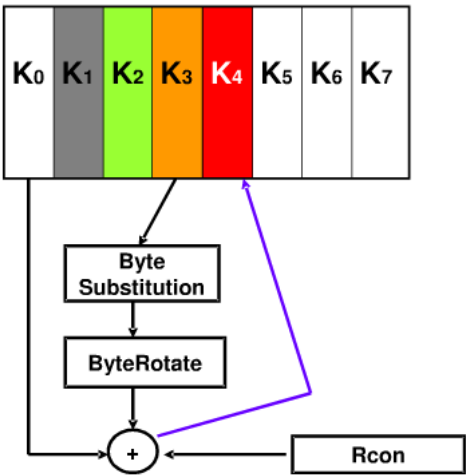
KeyExpansion（密钥扩展）

将输入的密钥扩展为11组128位密钥组，当中第0组为输入密钥本身

其后第n组第i列 为 第n-1组第i列 与 第n组第i-1列之和（模2加法，1<= i <=3）



对于每一组 第一列即*i*=0, 有特殊的处理



将前一列即第*n*-1组第3列的4个字节循环左移1个字节，
并对每一个字节进行字节替代变换SubBytes
将第一行（即第一个字节）与轮常量rc[n]相加
最后再与前一组该列相加

```
1 void
2 AES::KeyExpansion(unsigned char*
3   key, unsigned char
4   w[][4][4])
5 {
6   int
7   i, j, r, c;
8   unsigned
9   char
10  rc[] = {0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80, 0x1b, 0x36};
11
12  for(r=0;
13    r<4; r++)
```

```
11     {
12         for(c=0;
13         c<4; c++)
14         {
15             w[0][r][c]
16             = key[r+c*4];
17         }
18     }
19     for(i=1;
20     i<=10; i++)
21     {
22         for(j=0;
23         j<4; j++)
24         {
25             unsigned
26             char
27             t[4];
28             for(r=0;
29             r<4; r++)
30             {
31                 t[r]
32                 = j ? w[i][r][j-1] : w[i-1][r][3];
33             }
34             if(j
35             == 0)
36             {
37                 unsigned
38                 char
39                 temp = t[0];
40                 for(r=0;
41                 r<3; r++)
42                 {
43                     t[r]
44                     = Sbox[t[(r+1)%4]];
45                 }
46                 t[3]
47                 = Sbox[temp];
48                 t[0]
49                 ^= rc[i-1];
```

```

    }

    for(r=0;
r<4; r++)

    {

        w[i][r][j]
= w[i-1][r][j] ^ t[r];

    }

}

}

}

```

解密的基本运算

AES解密算法与加密不同，基本运算中除了AddRoundKey（轮密钥加）不变外，其余的都要进行逆变换，即

InvSubBytes（逆字节替代）、InvShiftRows（逆行移位）、InvMixColumns（逆列混淆）

```

1  void
2  AES::InvSubBytes(unsigned char
3  state[][4])
4  {
5      int
6      r,c;
7      for(r=0;
8      r<4; r++)
9      {
10         for(c=0;
11         c<4; c++)
12         {
13             state[r][c]
14             = InvSbox[state[r][c]];
15         }
16     }
17 }
18 void
19 AES::InvShiftRows(unsigned char
state[][4])

```

```
20 {
21     unsigned
22     char
23     t[4];
24     int
25     r,c;
26     for(r=1;
27         r<4; r++)
28     {
29         for(c=0;
30             c<4; c++)
31         {
32             t[c]
33             = state[r][(c-r+4)%4];
34         }
35         for(c=0;
36             c<4; c++)
37         {
38             state[r][c]
39             = t[c];
40         }
41     }
42 }
43
44 void
45 AES::InvMixColumns(unsigned char
46 state[][4])
47 {
48     unsigned
49     char
50     t[4];
51     int
52     r,c;
53     for(c=0;
54         c< 4; c++)
55     {
```

```

        for(r=0;
        r<4; r++)

        {

            t[r]
            = state[r][c];

        }

        for(r=0;
        r<4; r++)

        {

            state[r][c]
            = Ffmul(0x0e, t[r])

            ^

            Ffmul(0x0b, t[(r+1)%4])

            ^

            Ffmul(0x0d, t[(r+2)%4])

            ^

            Ffmul(0x09, t[(r+3)%4]);

        }

    }

}

```

加密过程

先将输入的明文按列序组合成4*4的矩阵，直接与第0组密钥（即输入的密钥）相加（异或），作为轮加密的输入

然后循环10次进行SubBytes、ShiftRows、MixColumns、AddRoundKey运算，最后恢复原序列
须要注意的是**最后一轮并不进行MixColumns**（列混淆变换）

```

1  unsigned
   char*
2  AES::Cipher(unsigned char*
   input)
3
4  {
5
6      unsigned
       char
7
8      state[4][4];
9
10     int
11
12     i, r, c;
13
14
15     for(r=0;
16     r<4; r++)
17
18     {

```

```
12         for(c=0;
13             c<4 ;c++)
14             {
15                 state[r][c]
16             = input[c*4+r];
17             }
18
19     AddRoundKey(state,w[0]);
20
21     for(i=1;
22         i<=10; i++)
23     {
24         SubBytes(state);
25         ShiftRows(state);
26         if(i!=10)MixColumns(state);
27         AddRoundKey(state,w[i]);
28     }
29
30     for(r=0;
31         r<4; r++)
32     {
33         for(c=0;
34             c<4 ;c++)
35         {
36             input[c*4+r]
37             = state[r][c];
38         }
39     }
40
41     return
42
43     input;
44 }
```

解密过程

```
1  unsigned
   char*
2  AES::InvCipher(unsigned char*
   input)
3
4  {
5      unsigned
   char
6      state[4][4];
7
8      int
   i, r, c;
9
10     for(r=0;
   r<4; r++)
11     {
12         for(c=0;
   c<4 ;c++)
13         {
14             state[r][c]
15             = input[c*4+r];
16         }
17     }
18
19     AddRoundKey(state,
20     w[10]);
21
22     for(i=9;
   i>=0; i--)
23     {
24         InvShiftRows(state);
25
26         InvSubBytes(state);
27
28         AddRoundKey(state,
29         w[i]);
30
31         if(i)InvMixColumns(state);
32     }
33
34     for(r=0;
   r<4; r++)
35     {
36         for(c=0;
   c<4 ;c++)
```

```
        {  
  
            input[c*4+r]  
            = state[r][c];  
  
        }  
  
    }  
  
    return  
  
    input;  
  
}
```

对外部数据的加密/解密

至此已经实现了AES加密与解密的原型，在使用的时候一般处理的是字符串等，而不是直接传入128位的数据，所以要封装一下对外部数据的加解密处理

```
1  void*  
   AES::Cipher(void*  
2  input, int  
3  length)  
4  {  
5      unsigned  
   char*  
6  in = (unsigned char*)  
   input;  
7  
   int  
8  
   i;  
9  
   if(!length)  
10  {  
11  
       while(*(in+length++));  
12  
       in  
13  = (unsigned char*)  
   input;  
14  
   }  
15  
   for(i=0;  
16  i<length; i+=16)  
17  {  
18  
       Cipher(in+i);  
19  
   }  
20  
   return  
21  input;
```



```
22 }
23
24 void*
25     AES::InvCipher(void*
26         input, int
27         length)
28 {
29     unsigned
30     char*
31     in = (unsigned char*)
32     input;
33
34     int
35
36     i;
37
38     for(i=0;
39         i<length; i+=16)
40     {
41
42         InvCipher(in+i);
43
44     }
45
46     return
47
48     input;
49 }
```

好文要顶

关注我

收藏该文

ywxwkaifa

关注 - 0

粉丝 - 21

+加关注

« 上一篇: [HDU 4293 Groups \(线性dp\)](#)
» 下一篇: [【Bootstrap3.0建站笔记三】AspNetPager分页，每一列都可排序](#)

posted @ 2014-07-08 18:49yxwkaifa 阅读(269) 评论(0) 编辑 收藏

[刷新评论](#) [刷新页面](#) [返回顶部](#)

注册用户登录后才能发表评论，请 [登录](#) 或 [注册](#)，[访问网站首页](#)。

- 最新IT新闻:
- 三星官翻Galaxy Note7R确定! 依然要卖5000元
 - 小米把店开到“顶级商圈”，有多大意义？
 - 《魔兽争霸3》1.28补丁上线：更新太给力
 - 腾讯推成长守护平台 小学生看到后沉默了
 - IT首富多才多艺！马云实力献唱京剧《红灯记》选段
- » 更多新闻...
- 最新知识库文章:
- 如何打好前端游击战
 - 技术文章的阅读姿势
 - 马拉松式学习与技术人员的成长性
 - 程序员的“认知失调”
 - 为什么有的人工作多年还是老样子
- » 更多知识库文章...