实验原理

一．AES加密算法简介

1997年1月2日，NIST（美国国家标准研究所）开始了征集DES的替代者的工作。该替代者就称为高级加密标准，即AES（Advanced Encryption Standard）。1997年9月12日正式发布了征集AES的公告，要求AES具有128比特的分组长度，并支持128，192和256比特的密钥长度，同时要求AES要能在全世界范围内免费得到。

到1998年6月15日已有21个算法提交给NIST。NIST在1998年8月20日的“第一次AES候选大会”上宣布了15个AES的候选算法。在1999年3月举行了“第二次AES候选大会”之后，NIST于1999年8月宣布有5个候选算法入围最后的决赛：MARS，RC6，Rijndael，Serpent和Twofish。

2000年4月举行了“第三次AES候选大会”。2001年2月28日，NIST宣布了关于AES的联邦信息处理标准的草案可供公众讨论。2001年11月26日，Rijndael被采纳，成为AES标准，并在2001年12月4日的联邦记录中作为FIPS197公布。

AES的候选算法主要依据以下三条原则进行评判：

- 安全性
- 代价
- 算法与实现特性

其中，算法的“安全性”最为重要，如果一个算法被发现是不安全的就不会再被考虑。“代价”是各种实现算法的计算效率（如速度、存储需求等），包括软件实现、硬件实现和智能卡实现。“算法与实现特性”主要包括算法灵活性、简洁性及其他因素。最后，五个进入决赛的算法都被认为是安全的。而Rijndael(中文音译为“荣代尔”)之所以最后当选是由于它集安全性、性能、效率、可实现性及灵活性于一体，被认为优于其他四个算法。

二．AES加密流程 动画演示

对于任意长度的明文，AES首先对其进行分组，每组的长度为128位。分组之后将分别对每个128位的明文分组进行加密。

对于每个128位长度的明文分组的加密过程如下：

（1）将128位AES明文分组放入状态矩阵中。

（2）AddRoundKey变换：对状态矩阵进行AddRoundKey变换，与膨胀后的密钥进行异或操作（密钥膨胀将在实验原理七中详细讨论）。

（3）10轮循环：AES对状态矩阵进行了10轮类似的子加密过程。前9轮子加密过程中，每一轮子加密过程包括4种不同的变换，而最后一轮只有3种变换，前9轮的子加密步骤如下：

- SubBytes变换：SubBytes变换是一个对状态矩阵非线性的变换；
- ShiftRows变换：ShiftRows变换对状态矩阵的行进行循环移位；
- MixColumns变换：MixColumns变换对状态矩阵的列进行变换；
- AddRoundKey变换：AddRoundKey变换对状态矩阵和膨胀后的密钥进行异或操作。

最后一轮的子加密步骤如下：

- SubBytes变换：SubBytes变换是一个对状态矩阵非线性的变换；
- ShiftRows变换：ShiftRows变换对状态矩阵的行进行循环移位；
- AddRoundKey变换：AddRoundKey变换对状态矩阵和膨胀后的密钥进行异或操作；

（4）经过10轮循环的状态矩阵中的内容就是加密后的密文。

AES的加密算法的伪代码如下。

```
Nb = 4                          // 状态矩阵的列数
Nr = 10                         // 加密的轮数
Cipher(byte in[4*Nb], byte out[4*Nb], word w[Nb*(Nr+1)])
begin
    byte state[4, Nb]
    state = in
    AddRoundKey(state, w[0, Nb-1])
    for round = 1 step 1 to Nr-1
        SubBytes(state)
        ShiftRows(state)
        MixColumns(state)
        AddRoundKey(state, w[round*Nb, (round+1)*Nb-1])
    end for
    SubBytes(state)
    ShiftRows(state)
    AddRoundKey(state, w[Nr*Nb, (Nr+1)*Nb-1])
    out = state
end
```

在AES算法中，AddRoundKey变换需要使用膨胀后的密钥，原始的128位密钥经过膨胀会产生44个字（每个字为32位）的膨胀后的密钥，这44个字的膨胀后的密钥供11次AddRoundKey变换使用，一次AddRoundKey使用4个字（128位）的膨胀后的密钥。

三．AES的分组过程

对于任意长度的明文，AES首先对其进行分组，分组的方法与DES相同，即对长度不足的明文分组后面补充0即可，只是每一组的长度为128位。

AES的密钥长度有128比特，192比特和256比特三种标准，其他长度的密钥并没有列入到AES联邦标准中，在下面的介绍中，我们将以128位密钥为例。

四．状态矩阵

状态矩阵是一个4行、4列的字节矩阵，所谓字节矩阵就是指矩阵中的每个元素都是一个1字节长度的数据。我们将状态矩阵记为State，State中的元素记为S_{ij}，表示状态矩阵中第i行第j列的元素。128比特的明文分组按字节分成16块，第一块记为“块0”，第二块记为“块1”，依此类推，最后一块记为“块15”，然后将这16块明文数据放入到状态矩阵中，将这16块明文数据放入到状态矩阵中的方法如图6-2-1所示。

块0	块4	块8	块12
块1	块5	块9	块13
块2	块6	块10	块14
块3	块7	块11	块15

图6-2-1　将明文块放入状态矩阵中

五．AddRoundKey变换

状态矩阵生成以后，首先要进行AddRoundKey变换，AddRoundKey变换将状态矩阵与膨胀后的密钥进行按位异或运算，如下所示。

$$[S'_{0,c}, S'_{1,c}, S'_{2,c}, S'_{3,c}] = [S_{0,c}, S_{1,c}, S_{2,c}, S_{3,c}] \oplus [W_{round*Nb+c}] \quad \text{for } 0 \leqslant c < Nb$$

其中，c表示列数，数组W为膨胀后的密钥，round为加密轮数，Nb为状态矩阵的列数。

它的过程如图6-2-2所示。

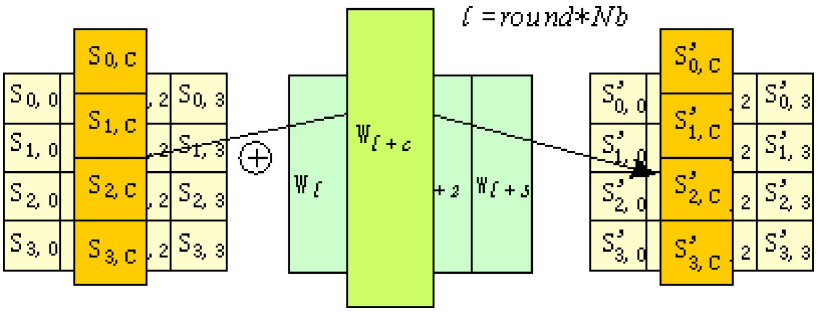

图6-2-2　AES算法AddRoundKey变换

六．10轮循环

经过AddRoundKey的状态矩阵要继续进行10轮类似的子加密过程。前9轮子加密过程中，每一轮要经过4种不同的变换，即SubBytes变换、ShiftRows变换、MixColumns变换和AddRoundKey变换，而最后一轮只有3种变换，即SubBytes变换、ShiftRows变换和AddRoundKey变换。AddRoundKey变换已经讨论过，下面分别讨论余下的三种变换。

1．SubBytes变换

SubBytes是一个独立作用于状态字节的非线性变换，它由以下两个步骤组成：

（1）在GF（2^8）域，求乘法的逆运算，即对于α∈GF（2^8）求β∈GF（2^8），使α β =β α = 1mod（$x^8 + x^4 + x^3 + x + 1$）。

（2）在GF（2^8）域做变换，变换使用矩阵乘法，如下所示：

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

由于所有的运算都在GF（2^8）域上进行，所以最后的结果都在GF（2^8）上。若g∈GF（2^8）是GF（2^8）的本原元素，则对于α∈GF（2^8），α≠0，则存在

β ∈ GF（2^8），使得：

β = g^{α} mod（$x^8 + x^4 + x^3 + x + 1$）

由于g^{255} = 1mod（$x^8 + x^4 + x^3 + x + 1$）

所以$g^{255-\alpha}$ = β^{-1}mod（$x^8 + x^4 + x^3 + x + 1$）

根据SubBytes变换算法，可以得出SubBytes的置换表，如表6-2-1所示，这个表也叫做AES的S盒。该表的使用方法如下：状态矩阵中每个元素都要经过该表替换，每个元素为8比特，前4比特决定了行号，后4比特决定了列号，例如求SubBytes(0C)查表的0行C列得FE。

表6-2-1　AES的SubBytes置换表

	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	01	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	B3	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	06	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

它的变换过程如图6-2-3所示。

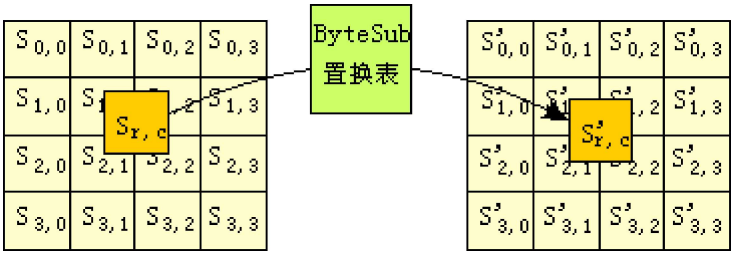

图6-2-3　SubBytes变换

AES加密过程需要用到一些数学基础，其中包括GF(2)域上的多项式、GF（2^8）域上的多项式的计算和矩阵乘法运算等，有兴趣的同学请参考相关的数学书籍。

2．ShiftRows变换

ShiftRows变换比较简单，状态矩阵的第1行不发生改变，第2行循环左移1字节，第3行循环左移2字节，第4行循环左移3字节。ShiftRows变换的过程如图6-2-4所示。

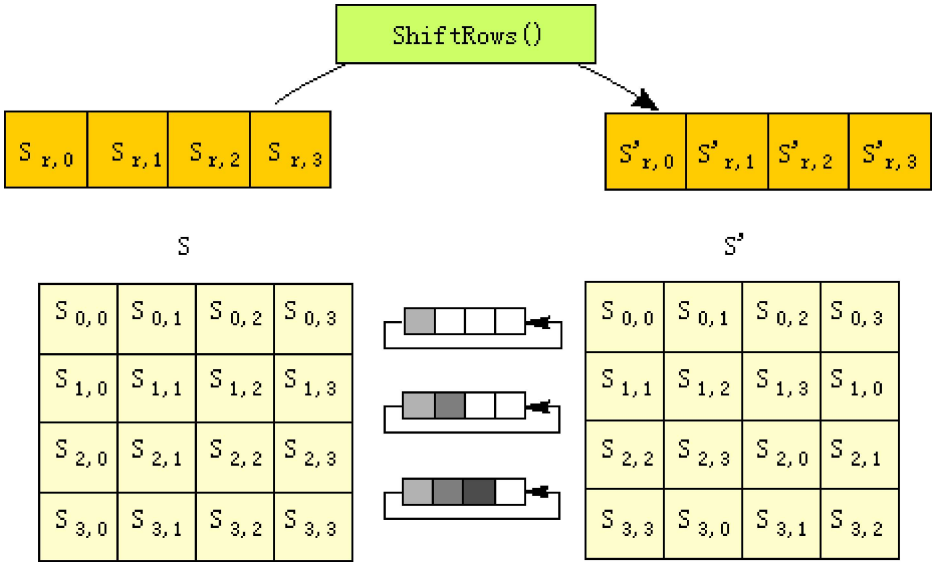

图6-2-4　AES的ShiftRows变换

3．MixColumns变换

在MixColumns变换中，状态矩阵的列看作是域GF（2^8）的多项式，模(x^4+1)乘以c（x）的结果：

$c(x)=(03)x^3+(01)x^2+(01)x+(02)$

这里(03)为十六进制表示，依此类推。c(x)与x^4+1互质，故存在逆：

$d(x)=(0B)x^3+(0D)x^2+(0G)x+(0E)$使$c(x)\bullet d(x) = (D1) \bmod (x^4+1)$。

设$b(x)=c(x)\otimes a(x)$有：

$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{bmatrix} 01 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix}$$

它的过程如图6-2-5所示。

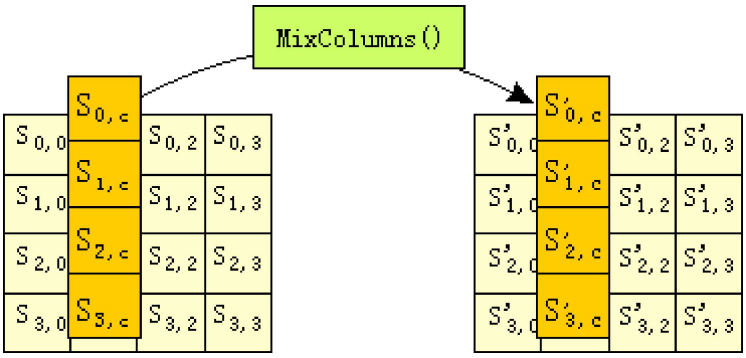

图6-2-5　AES算法MixColumns变换

七．密钥膨胀

在AES算法中，AddRoundKey变换需要使用膨胀后的密钥，膨胀后的密钥记为子密钥，原始的128位密钥经过膨胀会产生44个字（每个字为32位）的子密钥，这44个字的子密钥供11次AddRoundKey变换使用，一次AddRoundKey使用4个字（128位）的膨胀后的密钥。

密钥膨胀算法是以字为基础的（一个字由4个字节组成，即32比特）。128比特的原始密钥经过膨胀后将产生44个字的子密钥，我们将这44个密钥保存在一个字数组中，记为W[44]。128比特的原始密钥分成16份，存放在一个字节的数组：Key[0]，Key[1]……Key[15]中。

在密钥膨胀算法中，Rcon是一个10个字的数组，在数组中保存着算法定义的常数，分别为：

Rcon[0] = 0x01000000

Rcon[1] = 0x02000000

Rcon[2] = 0x04000000

Rcon[3] = 0x08000000

Rcon[4] = 0x10000000

Rcon[5] = 0x20000000

Rcon[6] = 0x40000000

Rcon[7] = 0x80000000

Rcon[8] = 0x1b000000

Rcon[9] = 0x36000000

另外，在密钥膨胀中包括其他两个操作RotWord和SubWord，下面对这两个操作做说明：

RotWord(B_0,B_1,B_2,B_3)对4个字节B_0,B_1,B_2,B_3进行循环移位，即

RotWord(B_0, B_1, B_2, B_3) = (B_1, B_2, B_3, B_0)

SubWord(B_0, B_1, B_2, B_3)对4个字节B_0, B_1, B_2, B_3使用AES的S盒，即

SubWord(B_0, B_1, B_2, B_3) = (B'_0, B'_1, B'_2, B'_3)

其中，B'_i = SubBytes（B_i），i = 0,1,2,3。

密钥膨胀的算法如下：

```
Nk = 4                  // 密钥长度，以字为单位（32 比特）
Nr = 10                 // 加密的轮数
Nb = 4                  // 状态矩阵的列数
KeyExpansion(byte key[4*Nk], work w[Nb*(Nr+1)], Nk)
begin
     word temp
     i = 0
     while(i < Nk)
          w[i] = word(key[4*i], key[4*i+1], key[4*i+2], key[4*i+3])
          i=i+1
     end while
     i = Nk
     while(i < Nb * (Nr+1))
          temp = w[i-1]
          if(i mode Nk = 0)
               temp = SubWord(RotWord(temp)) xor Rcon[i/Nk]
          else if(Nk > 6 and i mod Nk = 4)
               temp = SubWord(temp)
          end if
          w[i] = w[i-Nk] xor temp
          i = i + 1
     end while
end
```

八．解密过程

AES的加密和解密过程并不相同，首先密文按128位分组，分组方法和加密时的分组方法相同，然后进行轮变换。

AES的解密过程可以看成是加密过程的逆过程，它也由10轮循环组成，每一轮循环包括四个变换分别为InvShiftRows变换、InvSubBytes变换、InvMixColumns变换和AddRoundKey变换；

这个过程可以描述为如下代码片段所示：

```
Nk = 4                  // 状态矩阵的列数
Nr = 10                 // 加密的轮数

InvCipher(byte in[4*Nb], byte out[4*Nb], word w[Nb*(Nr+1)])
begin
     byte state[4,Nb]
     state = in
     AddRoundKey(state, w[Nr*Nb, (Nr+1)*Nb-1])
     for round = Nr-1 step -1 downto 1
          InvShiftRows(state)
          InvSubBytes(state)
          AddRoundKey(state, w[round*Nb, (round+1)*Nb-1])
          InvMixColumns(state)
     end for

     InvShiftRows(state)
     InvSubBytes(state)
     AddRoundKey(state, w[0, Nb-1])

     out = state
end
```

九．InvShiftRows变换

InvShiftRows变换是ShiftRows变换的逆过程，十分简单，指定InvShiftRows的变换如下。

$S_{r,(c+shift(r,Nb))modNb} = S_{r,c}$ for $0 < r < 4$ and $0 \leqslant c < Nb$

图6-2-6演示了这个过程。

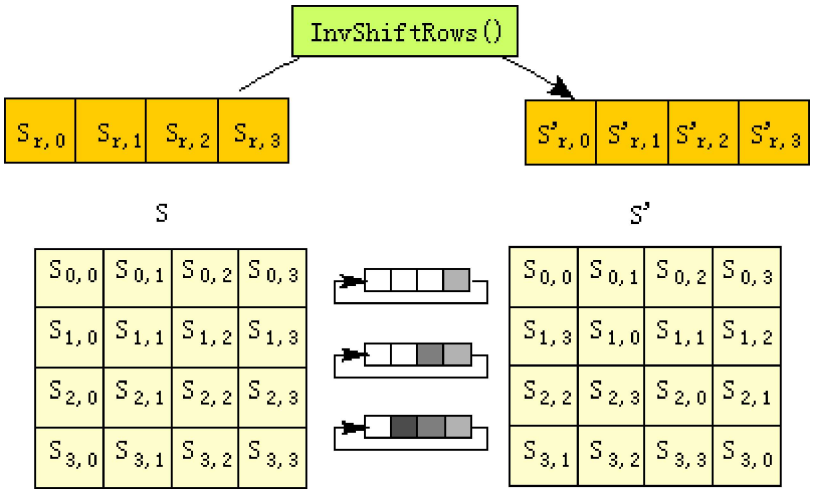

图6-2-6　AES算法InvShiftRows变换

十．InvSubBytes变换

InvSubBytes变换是SubBytes变换的逆变换，利用AES的S盒的逆作字节置换，表6-2-2为InvSubBytes变换的置换表。

表6-2-2　InvSubBytes置换表

	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

十一．InvMixColumns变换

InvMixColumns变换与MixColumns变换类似，每列乘以d(x)

$d(x) = (0B)x^3 + (0D)x^2 + (0G)x + (0E)$

下列等式成立：

$$((03)x^3 + (01)x^2 + (01)x + (02)) \odot d(x) = (01)$$

上面的内容可以描述为以下的矩阵乘法：

$$S'(x) = a^{-1}(x) \otimes s(x):$$

$$\begin{bmatrix} S'_{0,c} \\ S'_{1,c} \\ S'_{2,c} \\ S'_{3,c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} S_{0,c} \\ S_{1,c} \\ S_{2,c} \\ S_{3,c} \end{bmatrix} \text{for} 0 \leq c < Nb.$$

十二．AddRoundKey变换

AES解密过程的AddRoundKey变换与加密过程中的AddRoundKey变换一样，都是按位与子密钥做异或操作。解密过程的密钥膨胀算法也与加密的密钥膨胀算法相同。最后状态矩阵中的数据就是明文。