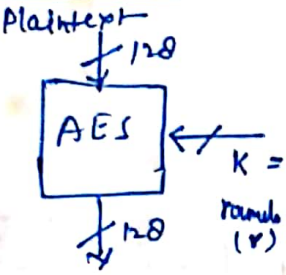
Advanced Encryption Standard (AES) (Block cipher) (Rijndael)

Rijman & Daemen



n = 128 bit block cipher

K = 128/192/256 ← depending on the size of key

rounds = 10/12/14
(r)

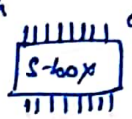
AES-128
AES-192
AES-256

in each each n = 64 fix

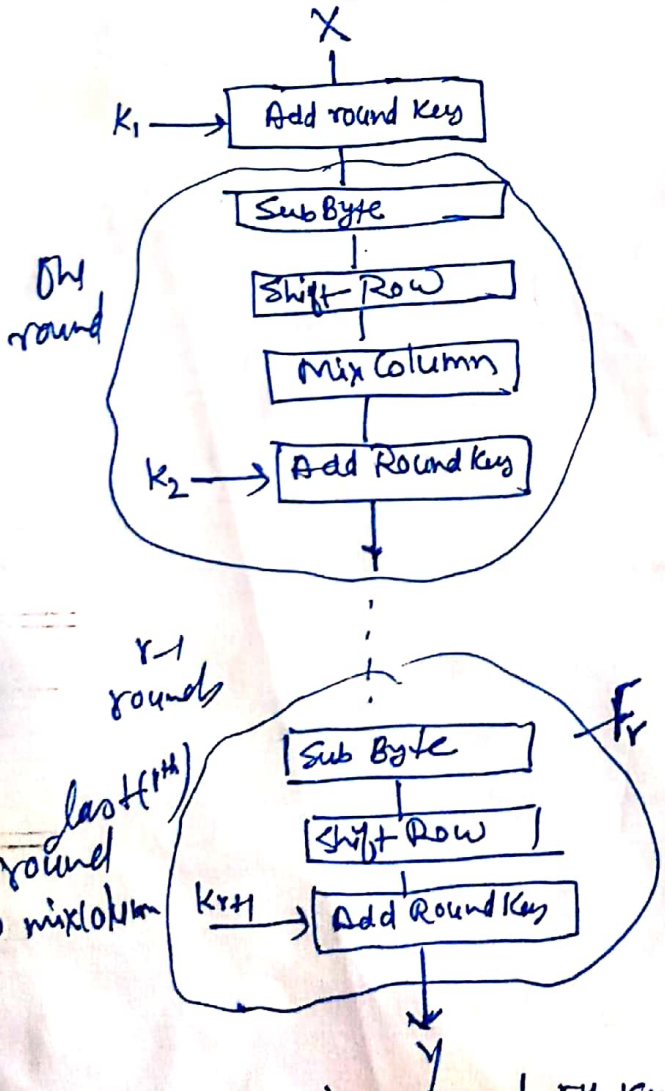
r-round, n-bit block cipher

Byte Operation

basic operations
- Add round key (bitwise XOR) with
- SubByte — S-box
- Shift-Row
- MixColumn

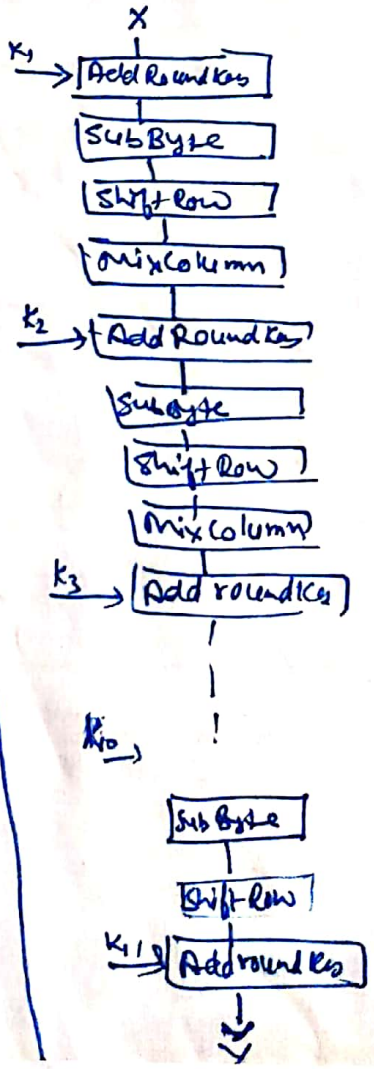


Then will continue for r-1 rounds
last round 1th mixcolumn is missing



One extra key in r round rth keys

AES-128 n = 128 K = 128 bit r = 10



प्रयोग/प्रोजेक्ट न० ..... दिनाँक .....

State 4×4 matrix — each entry is a byte (8 bit)

$$\begin{pmatrix} s_{00} & s_{01} & s_{02} & s_{03} \\ s_{10} & s_{11} & s_{12} & s_{13} \\ s_{20} & s_{21} & s_{22} & s_{23} \\ s_{30} & s_{31} & s_{32} & s_{33} \end{pmatrix} \leftarrow \begin{pmatrix} x_1 & x_5 & x_9 & x_{13} \\ x_2 & x_6 & x_{10} & x_{14} \\ x_3 & x_7 & x_{11} & x_{15} \\ x_4 & x_8 & x_{12} & x_{16} \end{pmatrix} \leftarrow | x_1 | x_2 | x_3 | x_4 | x_5 | \cdots | x_{16} |$$

State — 4×4 — $x_i \rightarrow$ Byte

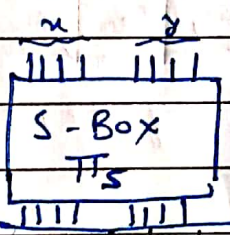
128 bit message converting into State

– **Add Round Key** – Bitwise XOR

$(101 \text{ ——})_{128 \text{ bit}}$ ⊕

Round Key $K_i \rightarrow (100 \text{ ——})_{128 \text{ bit}}$

– **SubByte operation** – taking a byte (8 bit input) using a table give byte (8 bit) output



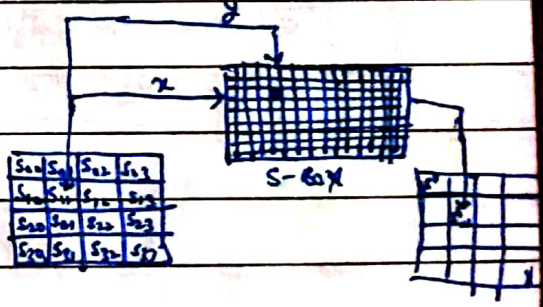
x, y are in hexadecimal form

4 bit 4 bit

Table in hexadecimal notation

x \ y	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	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	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

S-box in table form



Can also expressed in mathematical form using GField

टिप्पणी .....

अध्यापक हस्ता० : .....

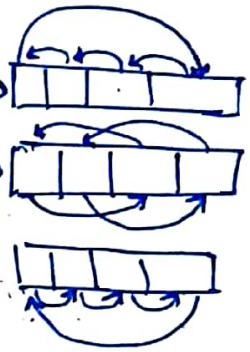
Subbyte

$$\begin{pmatrix} S_{00} & S_{01} & S_{02} & S_{03} \\ S_{10} & S_{11} & S_{12} & S_{13} \\ S_{20} & S_{21} & S_{22} & S_{23} \\ S_{30} & S_{31} & S_{32} & S_{33} \end{pmatrix} \Rightarrow \begin{pmatrix} \Pi_S(S_{00}) & \Pi_S(S_{01}) & & \\ & & & \\ & & & \\ & & & \end{pmatrix}$$

one state is converting into other using S-box table look up
this can also done using algebra

Shift Row

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



ShiftRows stage

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

First row no shifting
Second shift by 1
third ——— 2
fourth ——— 3

Mix Column operation

$$\begin{pmatrix} S_{00} & S_{01} & S_{02} & S_{03} \\ S_{10} & S_{11} & S_{12} & S_{13} \\ S_{20} & S_{21} & S_{22} & S_{23} \\ S_{30} & S_{31} & S_{32} & S_{33} \end{pmatrix} \Rightarrow \begin{pmatrix} S'_{00} & S'_{01} & S'_{02} & S'_{03} \\ S'_{10} & S'_{11} & S'_{12} & S'_{13} \\ S'_{20} & S'_{21} & S'_{22} & S'_{23} \\ S'_{30} & S'_{31} & S'_{32} & S'_{33} \end{pmatrix}$$

Here only a column of converting into a new column

प्रयोग/प्रोजेक्ट न० ..... दिनाँक .....

Algebraic formulation of AES S-box

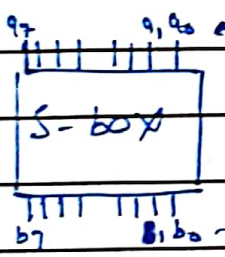
AES S-box involves operations in the finite field

$$F_{2^8} = Z_2[x]/(x^8+x^4+x^3+x+1)$$

FieldInv denote the multiplicative inverse of a field element. BinaryToField convert a byte to a field element and FieldToBinary perform the inverse conversion.

$$F_{2^8} = Z_2[x]/(x^8+x^4+x^3+x+1) = \{a_0+a_1x+a_2x^2+\cdots+a_7x^7 : a_i \in F_2\}$$

Here binary operation is multiplication modulo irreducible poly $x^8+x^4+x^3+x+1$



SubByte ($a_7 a_6 \cdots a_1 a_0$) input

1. $Z \leftarrow a_0+a_1x+\cdots+a_7x^7$ $\in F_{2^8}$
2. ($Z \neq 0$) (then it has inverse store it in Z)
   $Z \leftarrow$ Inverse(Z)
3. $Z = c_0+c_1x+\cdots+c_7x^7$ inverse is again a poly
4. ($a_7 a_6 \cdots a_1 a_0$) $\leftarrow$ ($c_7 c_6 \cdots c_1 c_0$)
5. ($c_7 c_6 \cdots c_1 c_0$) $\leftarrow$ (01100011)
6. $i \leftarrow 0$ to 7
7. $b_i \leftarrow (a_i + a_{i+4} + a_{i+5} + a_{i+6} + a_{i+7} + c_i) \bmod 2$
8. return ($b_7 b_6 \cdots b_0$)

$(a_0+a_1x+\cdots+a_7x^7)(b_0+\cdots+b_7x^7) \equiv 1 \pmod{x^8+x^4+x^3+x+1}$

These constant are given by designer adding $b = X(S_{ij})^{-1} + C$

Affine transformation matrix

$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \end{pmatrix} \oplus \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

Exm:- Suppose we have a byte {53} hexadecimal, which is 01010011

The corresponding field element is x^6+x^4+x+1

The multiplicative inverse (in F_{2^8}) can be shown to be

$$x^7+x^6+x^3+x$$

∴ in binary notation $(a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0) = (11001010)$

टिप्पणी ..... अध्यापक हस्ता० : .....

Next, compute $b_0 = a_0+a_4+a_5+a_6+a_7+c_0 \pmod 2$

$= 0+0+0+1+1+1 \pmod 2 = 1$

$b_1 = a_1+a_5+a_6+a_7+a_0+c_1 \pmod 2$

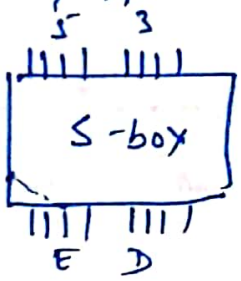
$= 1+0+1+1+0+1 \pmod 2 = 0$

⋮

$\therefore (b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0) = (1110\,1101)$ which is {ED} in hexadecimal notation

$\therefore$ If input is {53} output {ED} can be verified by table

$\therefore$ not only table lookup but also mathematical discription



Algebraic formulation of MixColumn — This is also a field operation

converting a column into new column

$$\begin{pmatrix} s_{00} & s_{01} & s_{02} & s_{03} \\ s_{10} & s_{11} & s_{12} & s_{13} \\ s_{20} & s_{21} & s_{22} & s_{23} \\ s_{30} & s_{31} & s_{32} & s_{33} \end{pmatrix} \Rightarrow \begin{pmatrix} s'_{00} \\ s'_{10} \\ s'_{20} \\ s'_{30} \end{pmatrix}$$

j^{th} Column

$j = 0,1,2,3$

$$\begin{pmatrix} s_{0j} \\ s_{1j} \\ s_{2j} \\ s_{3j} \end{pmatrix} \Rightarrow \begin{pmatrix} s'_{0j} \\ s'_{1j} \\ s'_{2j} \\ s'_{3j} \end{pmatrix}$$

$x \cdot$

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

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

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

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

s_{ij} is 8 bit $(a_7 a_6 \dots a_0) \Rightarrow a_0 + a_1 x + \dots + a_7 x^7$ multiply this field element by element x

This is field multiplication of XOR operation

प्रयोग/प्रोजेक्ट न० ..... दिनांक .....

Ex:- $S_{00} = \{87\}$ $S_{10} = \{6E\}$ $S_{20} = \{46\}$ $S_{30} = \{A6\}$

First Column

$x = 0000\,0010 = 02$ hexadecimal

$x+1 = 0000\,0011 = 03$ hexadecimal

$S'_{00} = (x \cdot S_{00}) \oplus (x+1)(S_{10}) \oplus S_{20} \oplus S_{30}$

$= (02 \cdot 87) \oplus (03 \cdot 6E) \oplus 46 \oplus A6 = 47$

To prove this $\{02\} = x$ $\{87\} = x^7 + x^2 + x + 1$

$(02 \cdot 87)$ $= x \cdot (x^7 + x^2 + x + 1) = x^8 + x^3 + x^2 + x$

$= x^4 + x^2 + 1 \pmod{x^8 + x^4 + x^3 + x + 1}$

$= 0001\,0101$ In Binary (Byte)

$\therefore$ $02 \cdot 87 = 0001\,0101$

Similarly $03 \cdot 6E = 1011\,0010$

$46 = 0100\,0110$

$A6 = 1010\,0110$

$\oplus\ 0100\,0111 = \{47\} = S'_{00}$

→ Similarly we can get $S'_{10}, S'_{20}, S'_{30}$

$$\begin{pmatrix} S_{00} \\ S_{10} \\ S_{20} \\ S_{30} \end{pmatrix} \Rightarrow \begin{pmatrix} S'_{10} \\ S'_{10} \\ S'_{20} \\ S'_{30} \end{pmatrix}$$

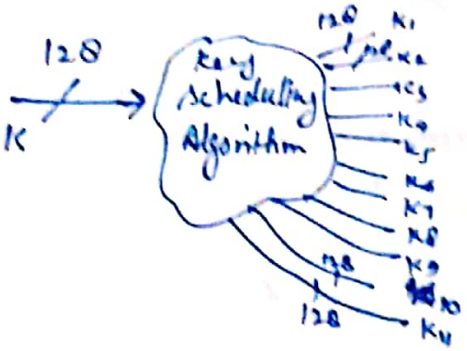
The transformation can be determined by the following matrix multiplication on State

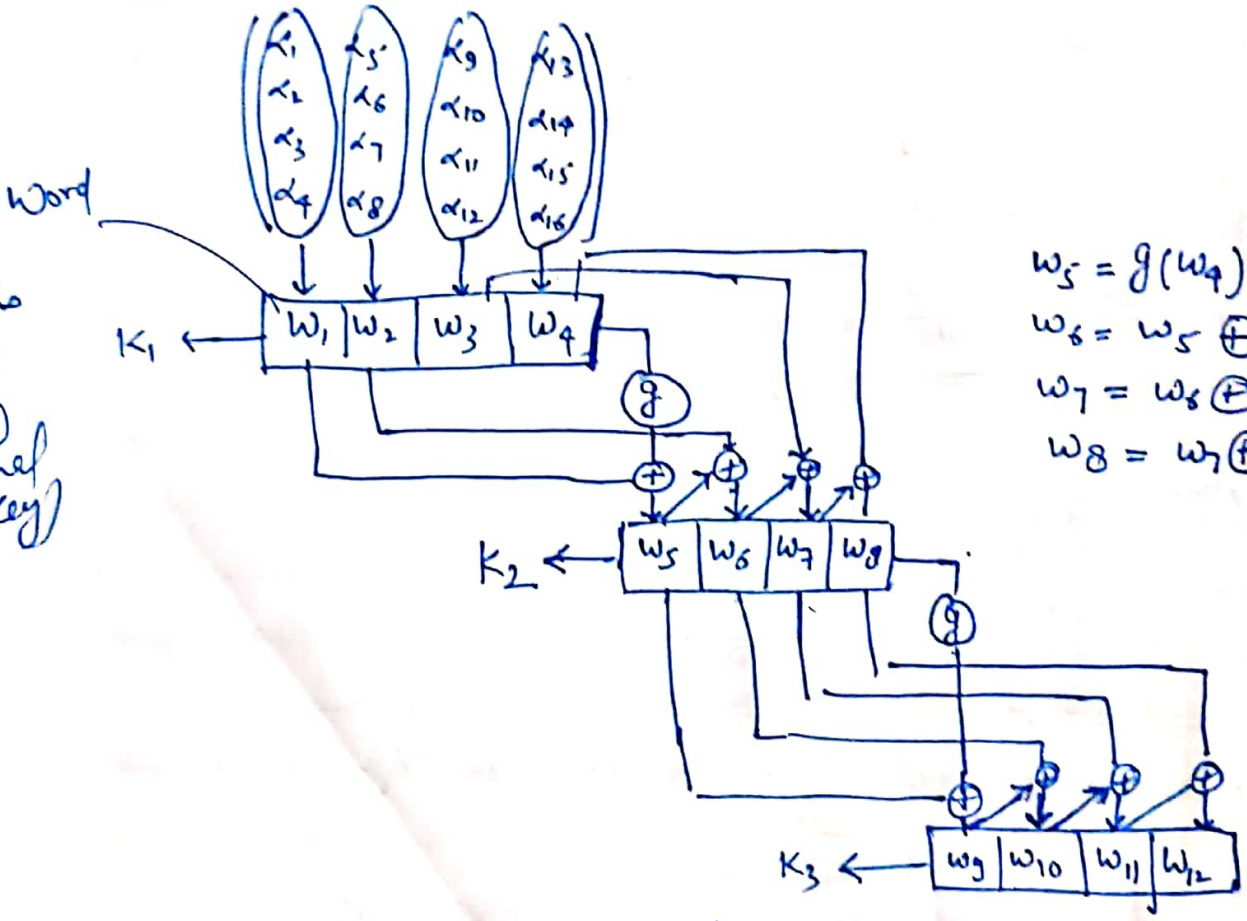
Hexa decimal notation

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} S_{00} & S_{01} & S_{02} & S_{03} \\ S_{10} & S_{11} & S_{12} & S_{13} \\ S_{20} & S_{21} & S_{22} & S_{23} \\ S_{30} & S_{31} & S_{32} & S_{33} \end{bmatrix} = \begin{bmatrix} S'_{00} & S'_{01} & S'_{02} & S'_{03} \\ S'_{10} & S'_{11} & S'_{10} & S'_{13} \\ S'_{20} & S'_{21} & S'_{22} & S'_{23} \\ S'_{30} & S'_{31} & S'_{32} & S'_{33} \end{bmatrix}$$

टिप्पणी ..... अध्यापक हस्ता० : .....

Key Scheduling Algorithm for AES (Give us round keys)





Similarly $K_4, \ldots, K_{11}$

$w_9 = g(w_8) \oplus w_5$
$w_{10} = w_9 \oplus w_6$
$w_{11} = w_{10} \oplus w_7$
$w_{12} = w_{11} \oplus w_8$

g fn. takes 32 bit input and give 32 bit output.

g fn. consists of following subfunctions:

1. RotWord performs a one-byte circular left shift on a word. This means an input word $[b_0\ b_1\ b_2\ b_3]$ is transformed into $[b_1\ b_2\ b_3\ b_0]$

2. SubWord performs a byte substitution on each byte of its input word, using the S-box described earlier

3. Result of step 1 & 2 is XOR with round constant Rcon[j]

g
$(b_0\ b_1\ b_2\ b_3)$
- circular left shift
- S-box
- XOR of add constant Rcon[j]

Rcon[1] ← 01 00 00 00
Rcon[2] ← 02 00 00 00
Rcon[3] ← 04 00 00 00
Rcon[4] ← 08 00 00 00
Rcon[5] ← 10 00 00 00
Rcon[6] ← 20 00 00 00
Rcon[7] ← 40 00 00 00
Rcon[8] ← 80 00 00 00
Rcon[9] ← 1B 00 00 00
Rcon[10] ← 36 00 00 00

Decryption of AES: