Yousef Almesbahi 16952013 计科 2

Advanced Encryption Standard

NOTE：Last code I have handed in I found that it misses the key schedule, which is the function for creating the new key.

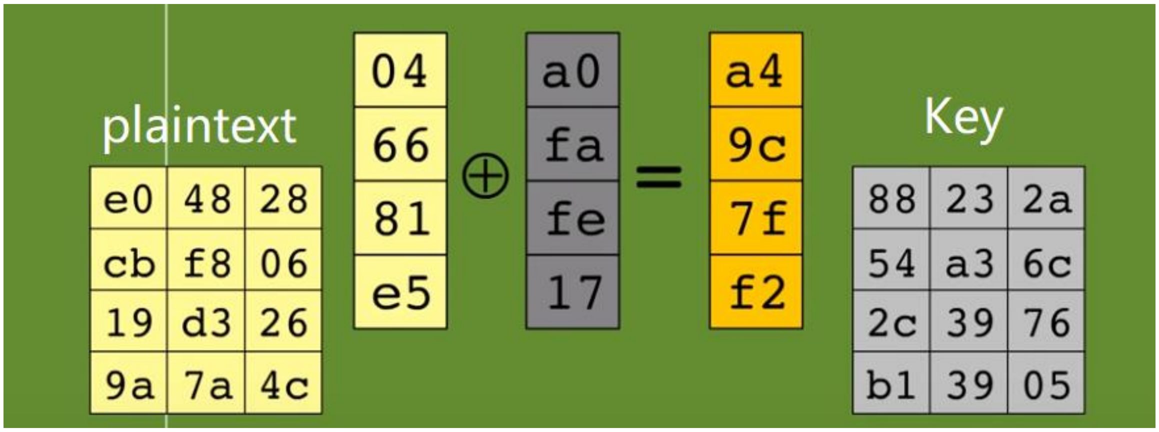
Algorithm:

- This is a 128 bits AES Rijndael cipher.
- The AES is a developed type of DES after finding some leakage problem in DES cipher.
- The text must be translated to hex as a matrix.
- It conation the following algorithms:
  - Key Expansion
  - Add round key
  - Shift row
  - Sub bytes
  - Mix column
- Every single need to be repeated almost 10 times for the 128 bits AES.
- Decrypting is doing the same steps of encrypting inversely.

Process:

- After typing in the plaintext and key or leaving the default plaintext and key they will be transferred to hexadecimal numbers and then from block to state (matrix).

1. **Add round key:**
   a. In this process we take the giving table of the key and generate the new text plain by XOR the plaintext state with the key state



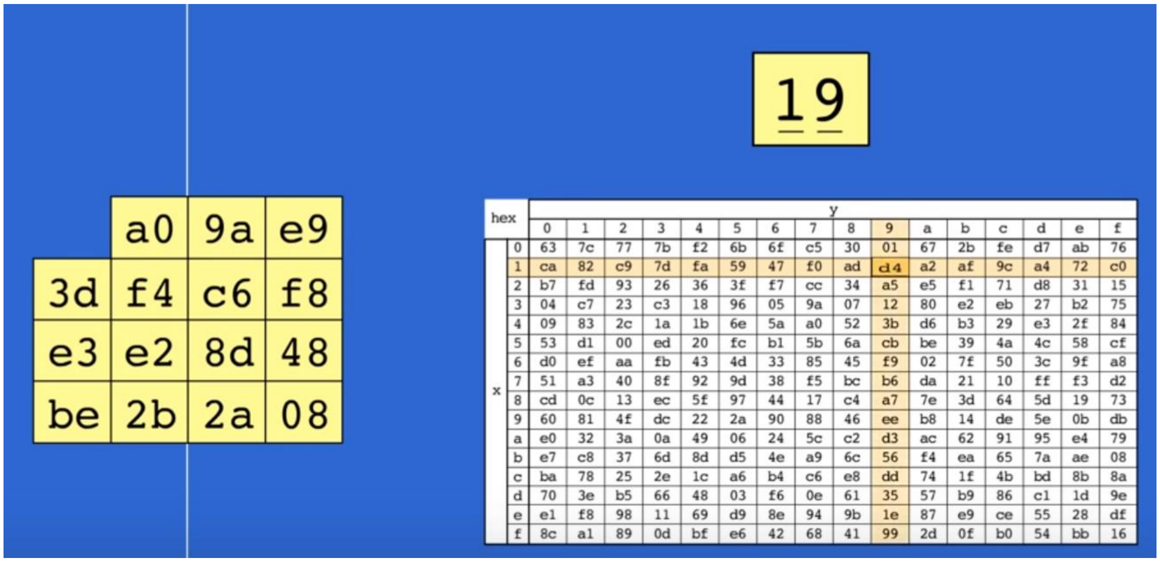
e0	48	28
cb	f8	06
19	d3	26
9a	7a	4c

04	⊕	a0	=	a4
66		fa		9c
81		fe		7f
e5		17		f2

88	23	2a
54	a3	6c
2c	39	76
b1	39	05

- Then we enter a loop of range of nine.

2. **Sub bytes:**
    a. In the sub bytes process we take all of the elements in the plain text state, look for it matching value in the S-Box table, and change the value of the plain text state.
    b. In decryption, we just need to use the inverse of S-Box.



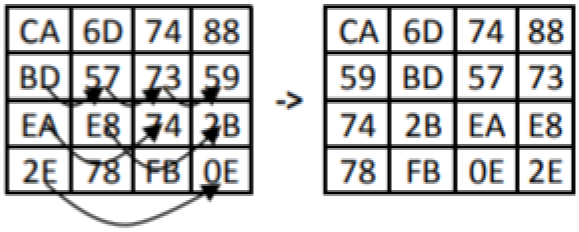
	a0	9a	e9
3d	f4	c6	f8
e3	e2	8d	48
be	2b	2a	08

19

hex		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	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

    c. In my code to use the inverse of S-Box you have to call the function with the parameter decryption = true.
    d. S-box constructed using defined transformation of values in Galois Field- GF(2^8)
    e. For every element in this state, we take the value of the box, and divide to two value left and right, left stand for the row number and right stand for the column number.

3. **Shift Rows:**
    a. A circular byte shift in each:
        i. 1st row is unchanged
        ii. 2nd row does 1 byte circular shift to left
        iii. 3rd row does 2 byte circular shift to left
        iv. 4th row does 3 byte circular shift to left
    b. For the decryption I just did the opposite way by shifting left 4 times the first line and one for the last line

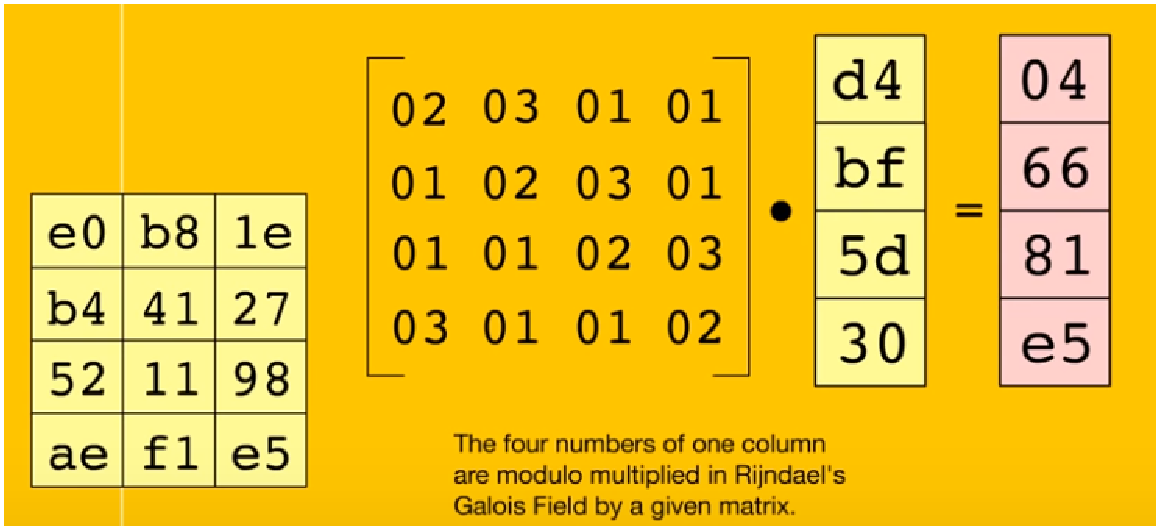


CA	6D	74	88
BD	57	73	59
EA	E8	74	2B
2E	78	FB	0E

->

CA	6D	74	88
59	BD	57	73
74	2B	EA	E8
78	FB	0E	2E

4. **MIX Columns:**
    a. In this state, we take the plaintext and multiply it with a **C** box of value.
    b. Effectively a matrix multiplication in GF(2^8) using prime poly m(x) =$x^8+x^4+x^3+x+1$
    c. In here I spend along trying to discover how to multiply these two matrices then I found a way online that teach how to simplify this method :
        i. That method teach how to simplify the C box or C^{-1} box element, the method is all about how to find the number from the C box with only multiplying by two and adding ones, for example (x * 3) = (x * 2) + x is 3x, here is a more detailed example:
            - (d4 ×02)+(bf ×03)+(5d ×01)+(30 ×01)
            - d4 ×02 is d4<<1, XOR with 1b (because the high bit of d4 is set), giving b3;
            - bf ×03 is bf<<1 XOR with 1b (because the high bit of bf is set) and bf(because we're multiplying by 3), giving us da;
            - 5d ×01 is 5d, and 30 ×01 is 30.
            - Now, we XOR b3, da, 5d and 30 together, and that gives us 04.
        ii. For 09, 0b, 0d and 0e here is the solution:
            - x ×9=(((x ×2) ×2) ×2)+x
            - x ×11=((((x ×2) ×2)+x) ×2)+x
            - x ×13=((((x ×2)+x) ×2) ×2)+x
            - x ×14=((((x ×2)+x) ×2)+x) ×2
    d. This go all through the matrix column by column.



e0	b8	1e
b4	41	27
52	11	98
ae	f1	e5

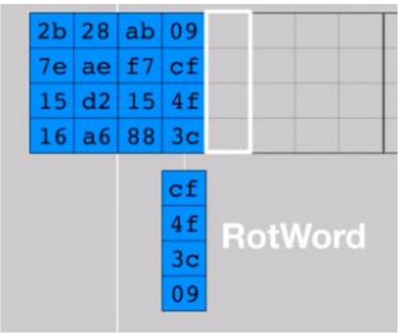
$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \bullet \begin{bmatrix} d4 \\ bf \\ 5d \\ 30 \end{bmatrix} = \begin{bmatrix} 04 \\ 66 \\ 81 \\ e5 \end{bmatrix}$$

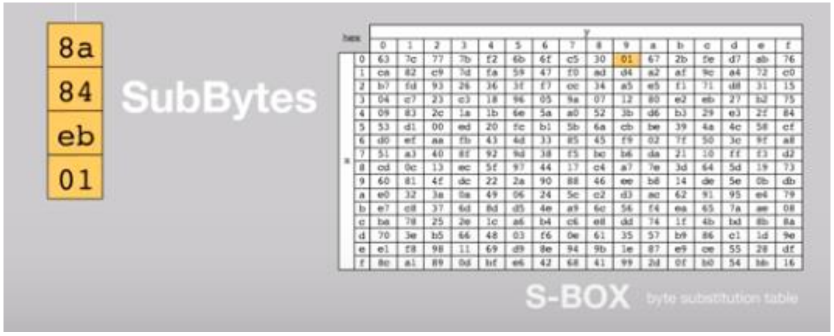
The four numbers of one column are modulo multiplied in Rijndael's Galois Field by a given matrix.

5. **Key schedule:**
    a. At end we have, the key scheduling which is has two formulas, first one for the first col, and the other three columns take different rule we call the original key as W0 and the new key as W1.

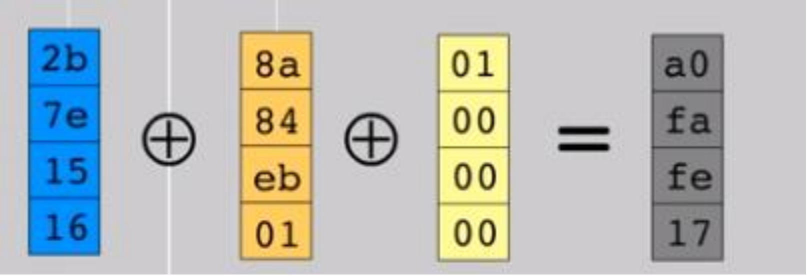
b. for the $W_0$1:

i. For $w_0$1 we should first get $W_3$0 and shift it one element to the left as illustrated, then we change all the from the S-Box in encryption and decryption.



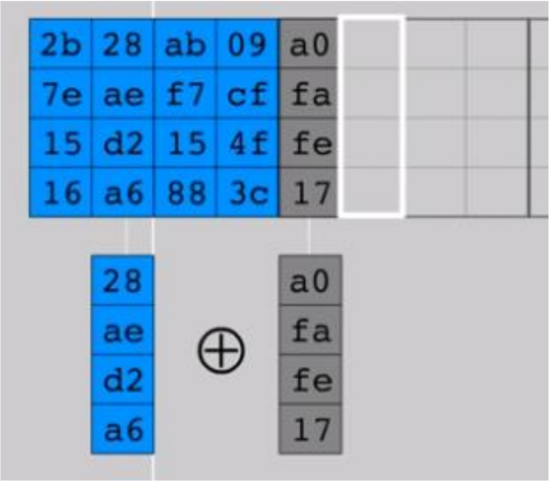


ii. Then we XOR the result with $W_0$0 and Rcon (Row constant), Rcon depends on the cycling of the repetitive functions, we start with zero and finish at nine for encryption and for decryption we do it the from nine to zero.



c. For $W_1$1, $W_2$1 and $W_3$1:

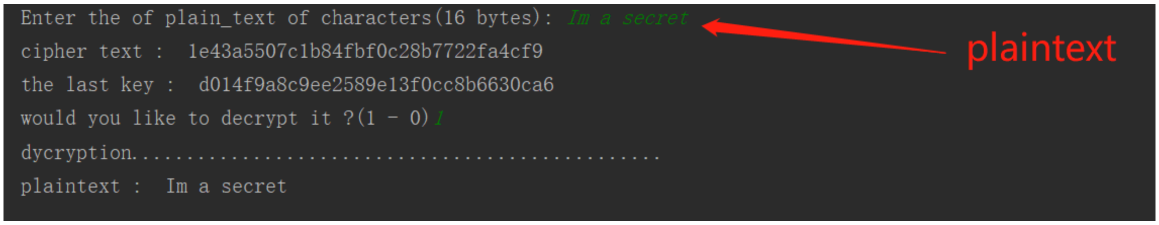
i. This is more simpler, we just need to XOR the W_{i-1}1 with W_i0 here is a picture for $W_1$1:



6. **The loop:**
   - **a.** Encryption:
     - **i. Add round key**
     - **ii.** Right now we have to repeat the above steps for nine times in this order:
       - **Sub bytes**
       - **Shift rows**
       - **Mix columns**
       - **Key schedule**
       - **Add round key**
     - **iii.** After the ninth loop we implement the following:
       - **Sub bytes**
       - **Shift rows**
       - **Key schedule**
       - **Add round key**
   - **b.** Decryption:
     - **i.** First we implement these:
       - **Add round key**
       - **Inverse shift row**
       - **Sub bytes**
       - **Key schedule inverse**
       - **Add round key**
     - **ii.** Then we start nine loops from nine to zero:
       - **Mix columns**
       - **Inverse shift row**
       - **Sub bytes**
       - **Key schedule inverse**
       - **Add round key**
7. **From state to block:**
   - **a.** Lastly, we change the state back to block and translate the hex to string to get the plaintext.

Result:

- The key and the plaintext can be set by default in the code, in the following result the key was set to default.



```
Enter the of plain_text of characters(16 bytes): Im a secret
cipher text :  1e43a5507c1b84fbf0c28b7722fa4cf9
the last key :  d014f9a8c9ee2589e13f0cc8b6630ca6
would you like to decrypt it ?(1 - 0) 1
dycryption..............................................
plaintext :  Im a secret
```