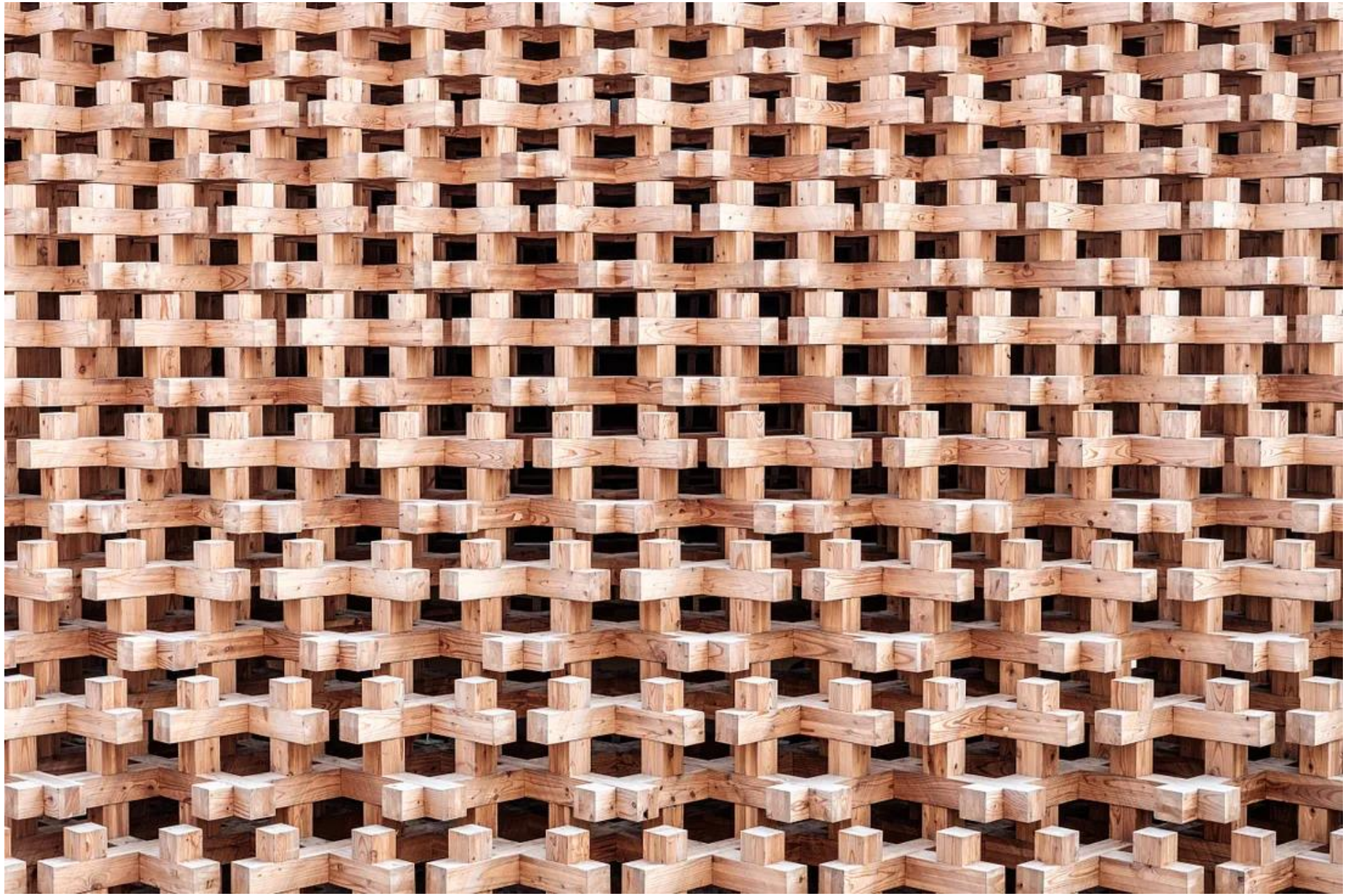
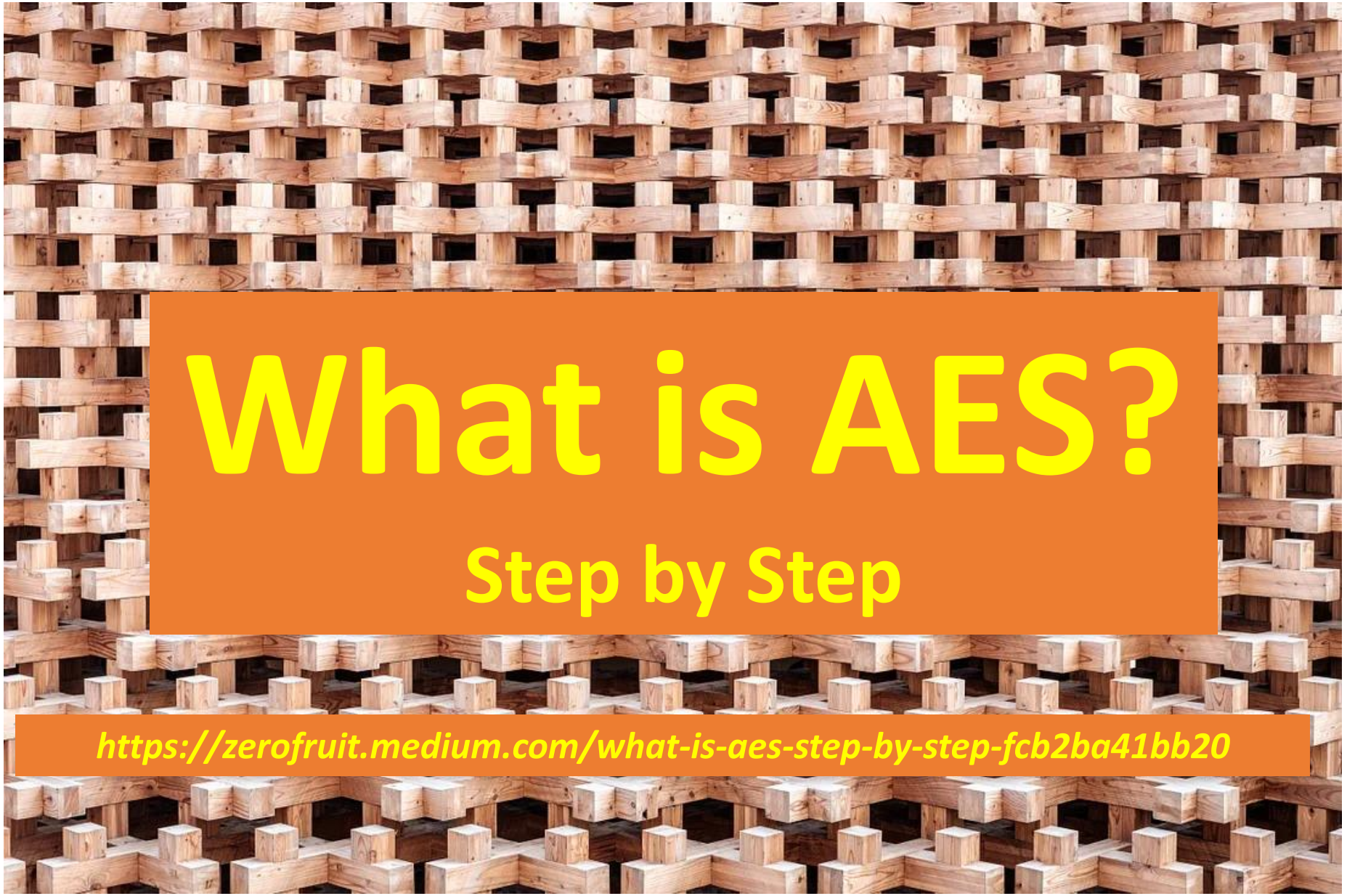
Advanced Encryption Standard

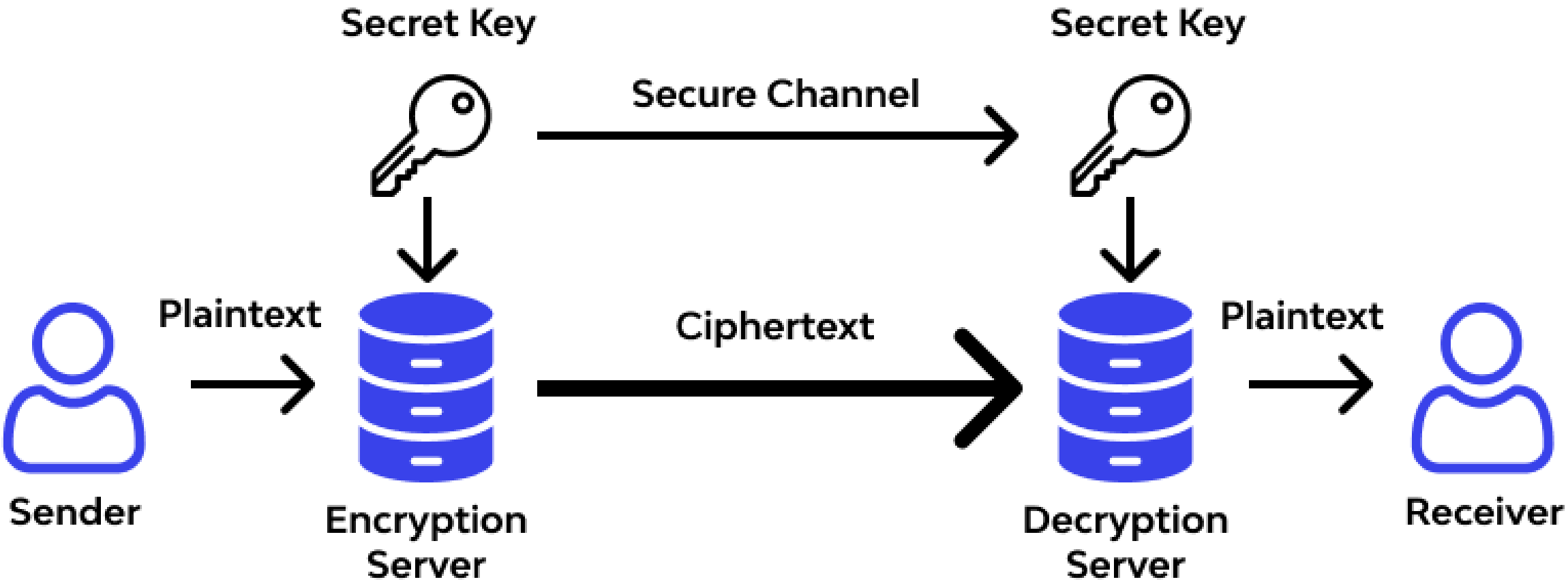
Advanced Encryption Standard



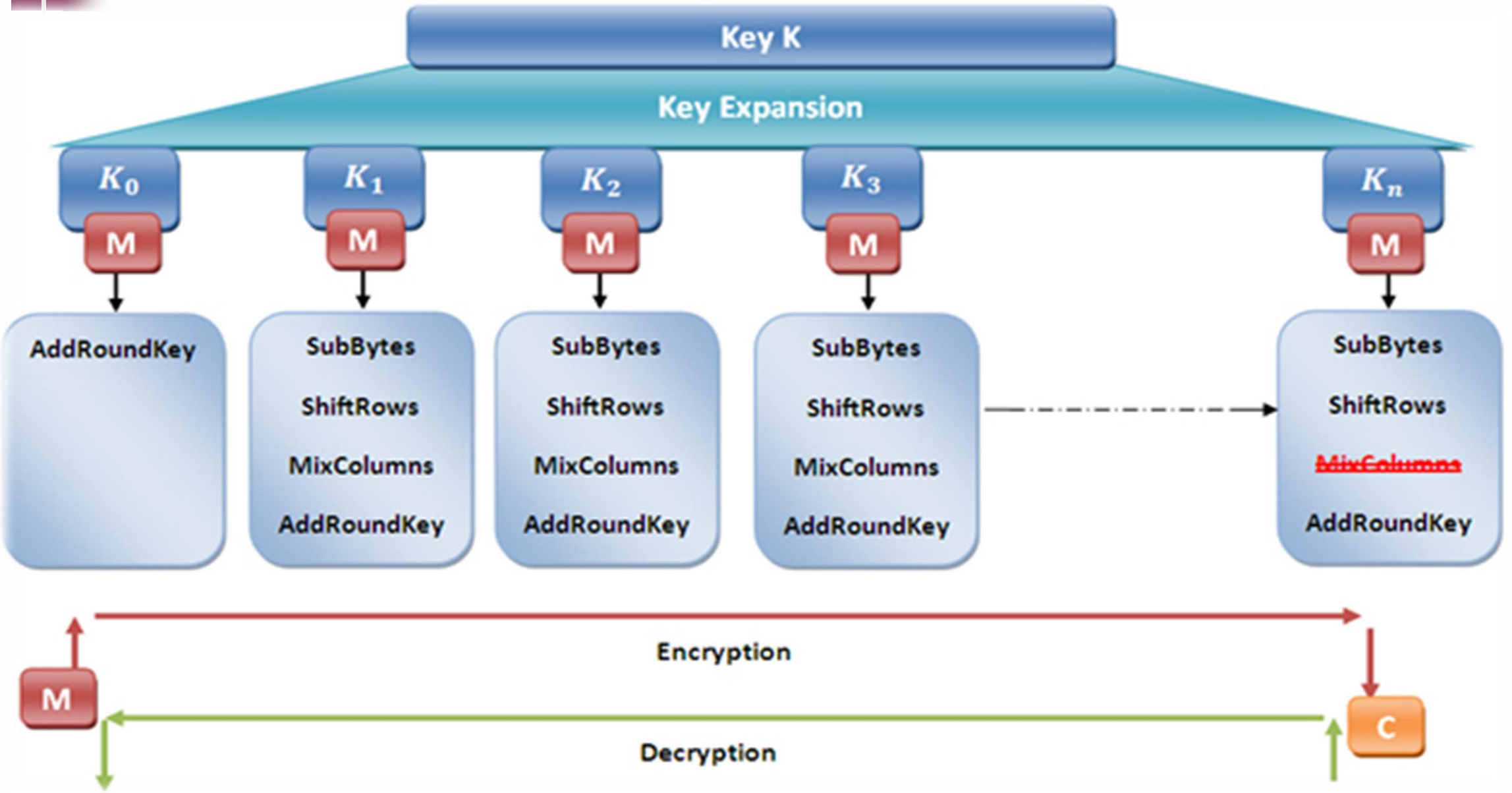
Cryptography AZ



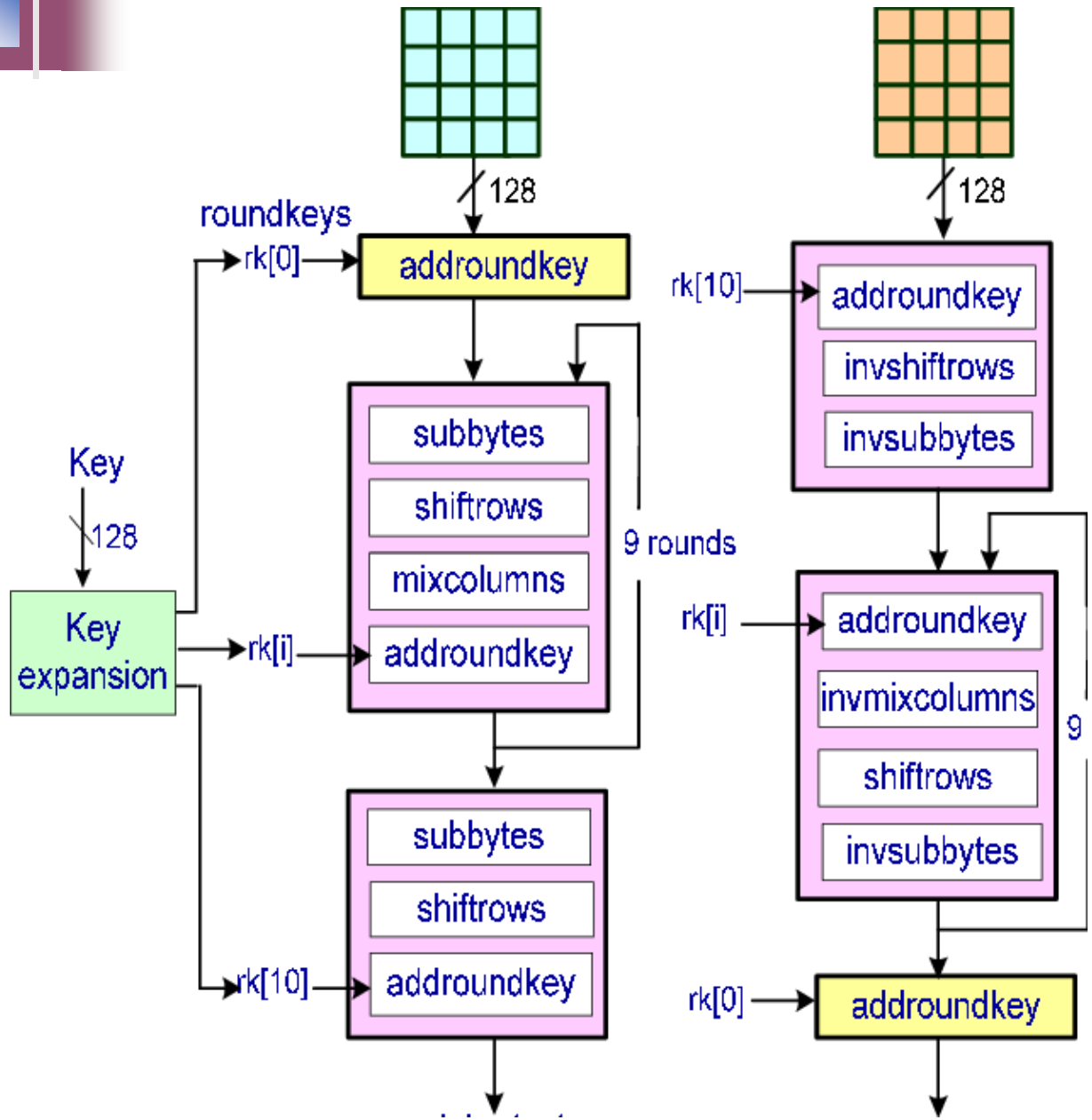
Advanced Encryption Standard

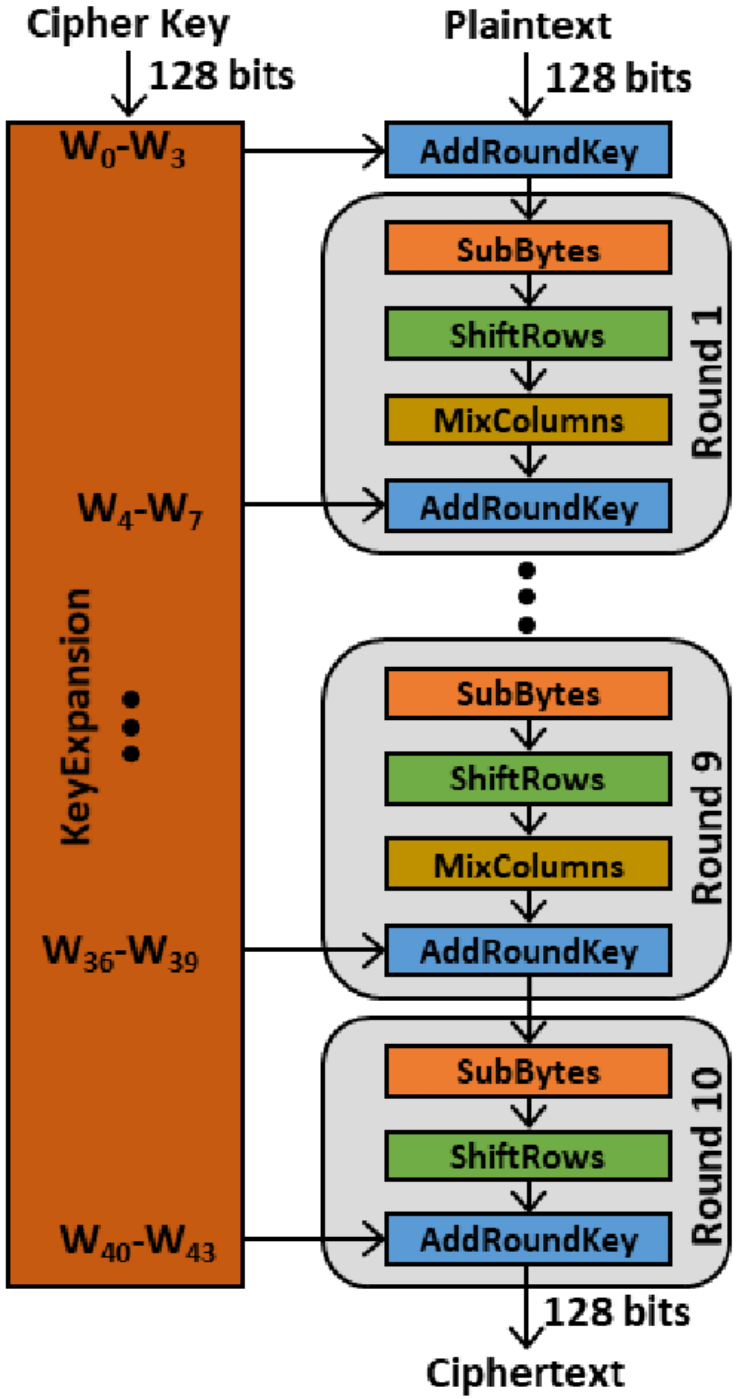


Advanced Encryption Standard (AES)

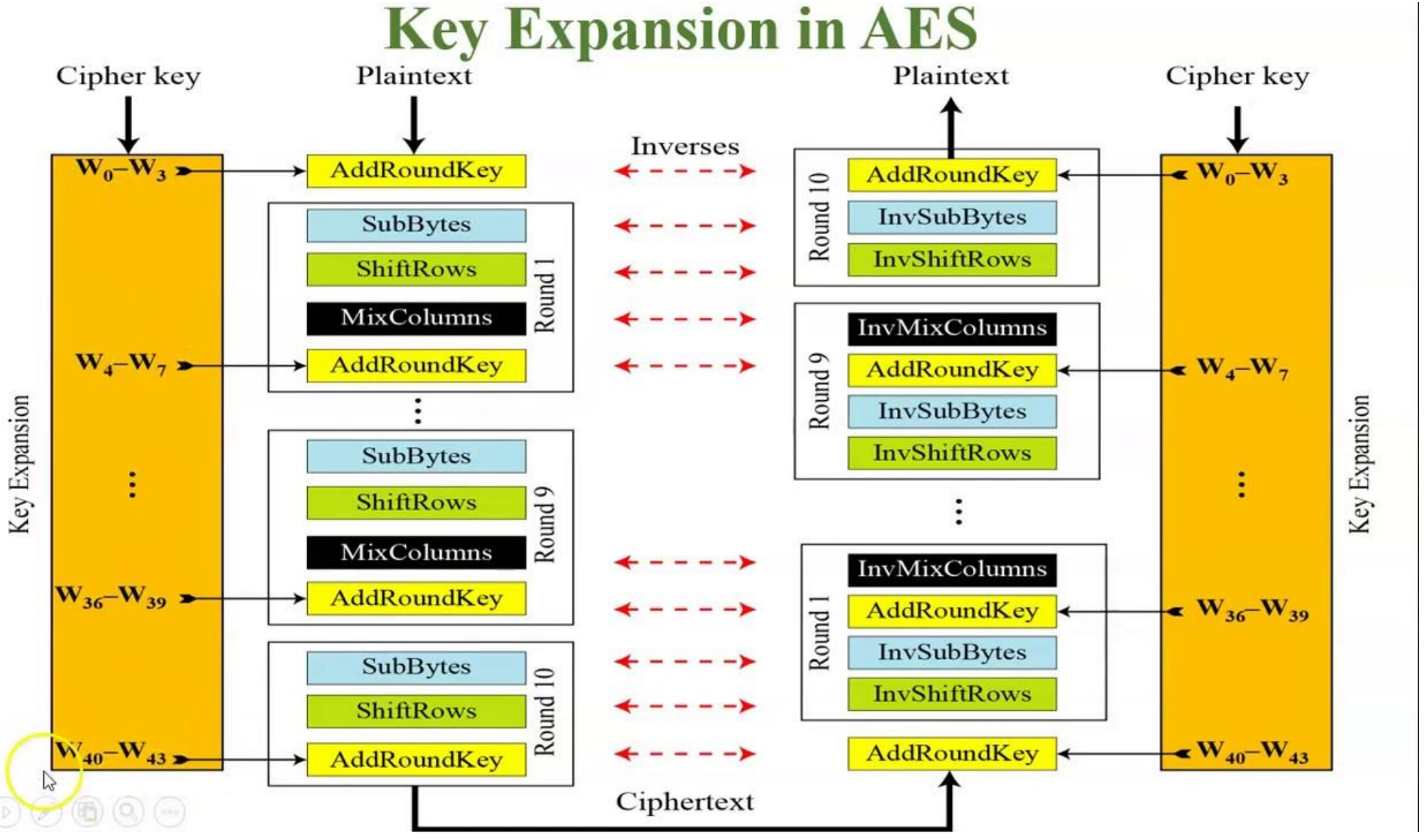


Advanced Encryption Standard (AES)

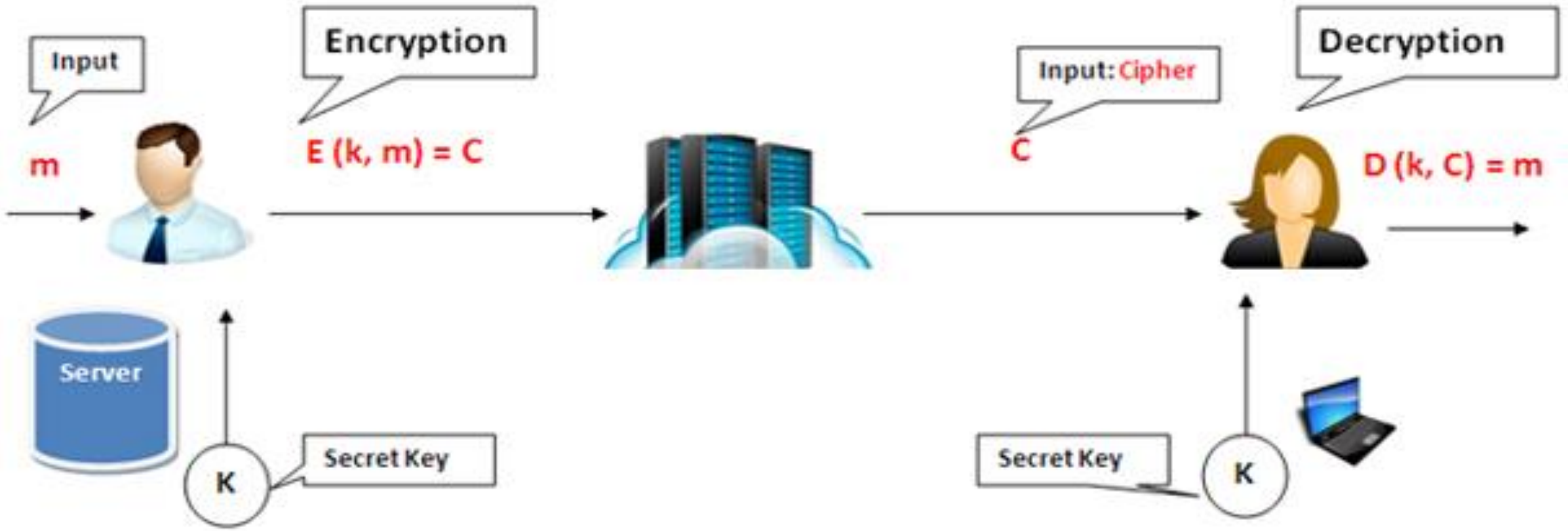




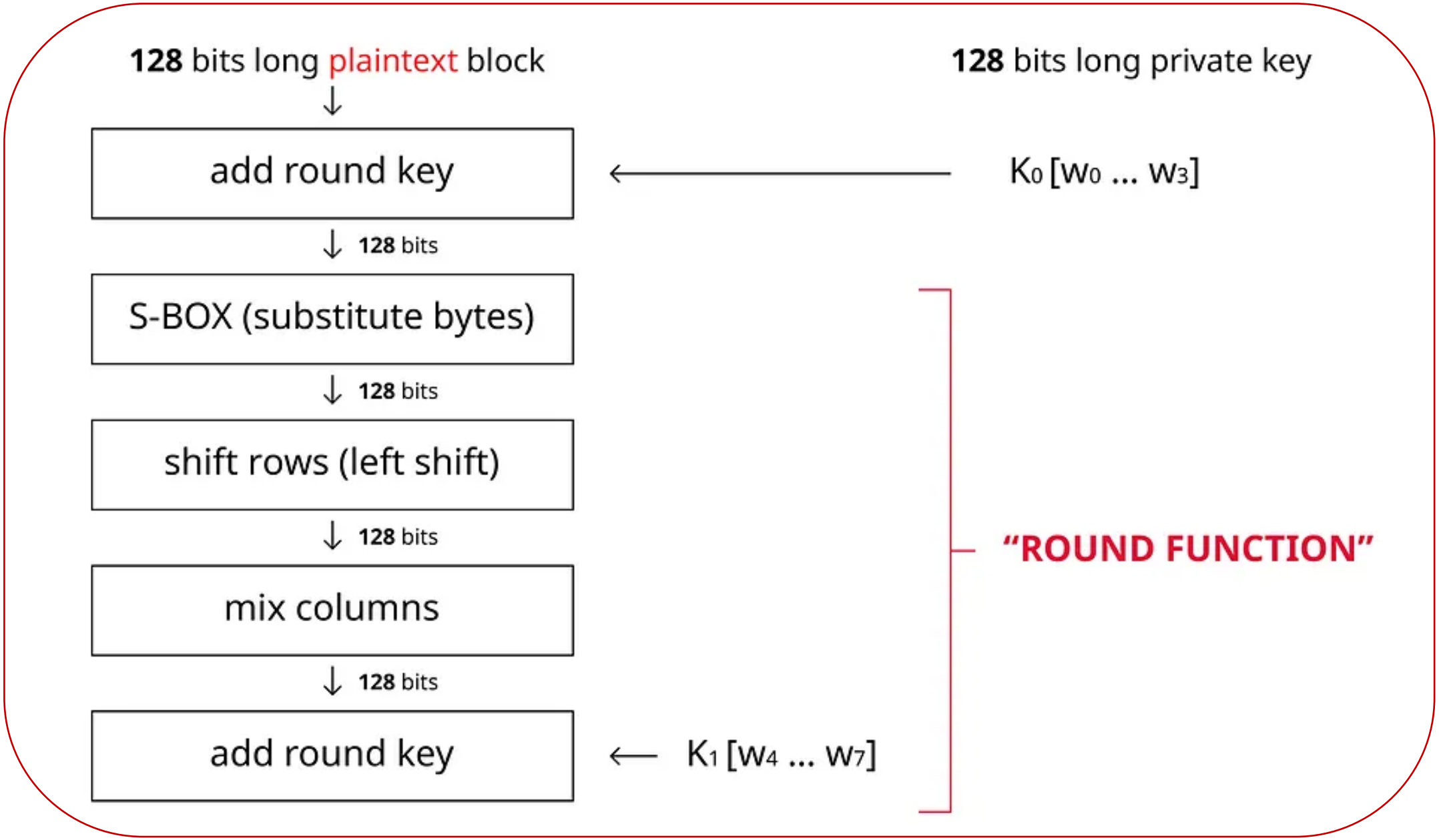
Advanced Encryption Standard (AES)



Advanced Encryption Standard (AES)



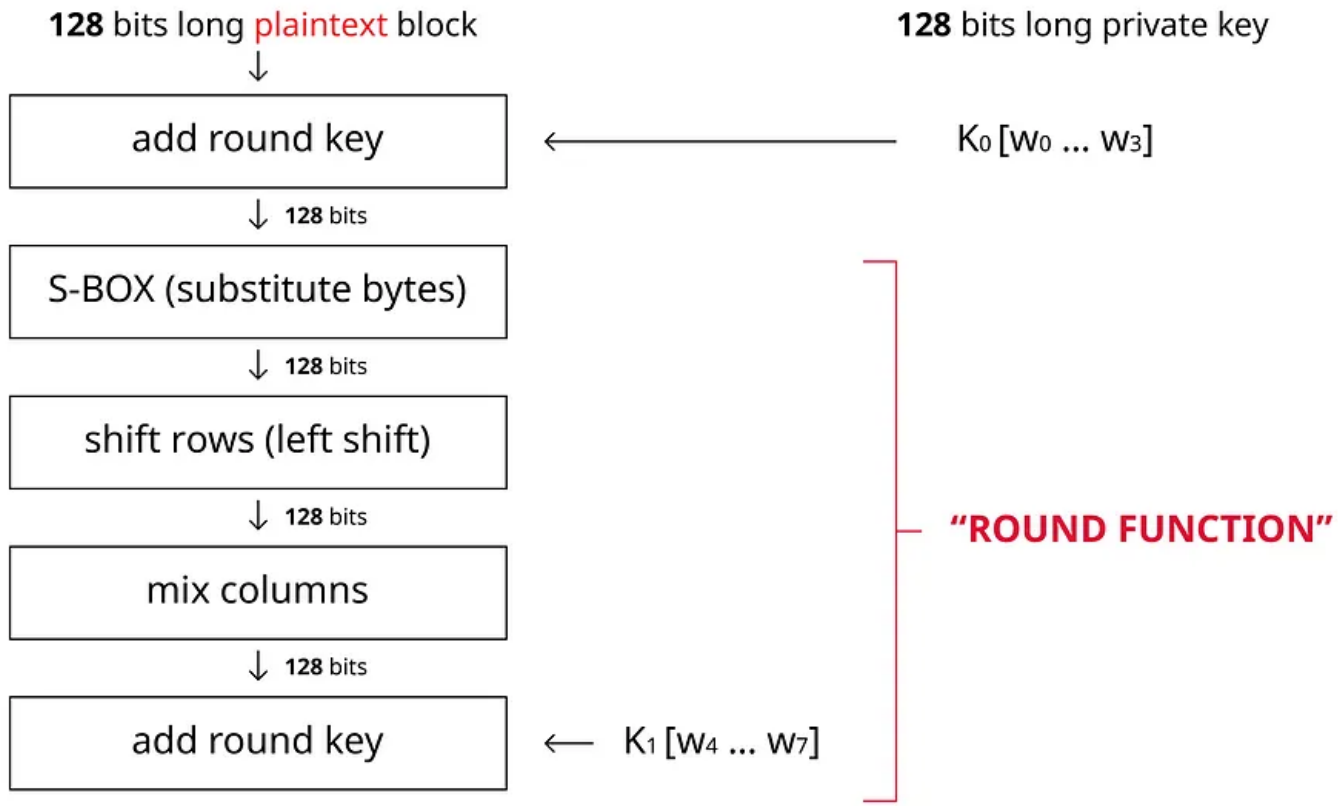
Advanced Encryption Standard (AES)



AES CIPHER

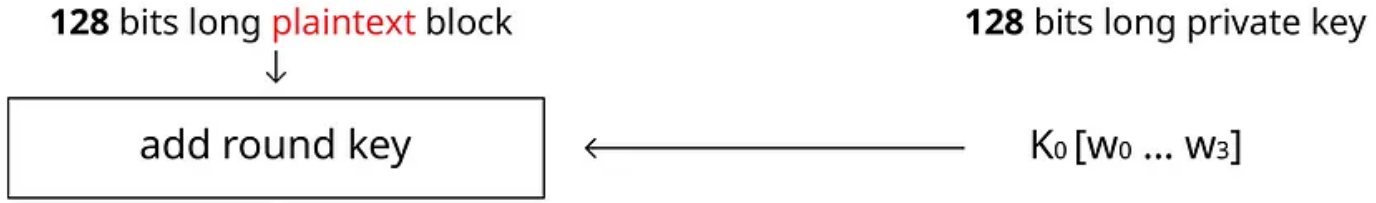
- Represent data as matrixes
- Add round key
- Round function
  - Substitute bytes
  - Shift rows
  - Mix columns
  - Add round key

- subkey generation



AES CIPHER

- **Represent data as matrixes**



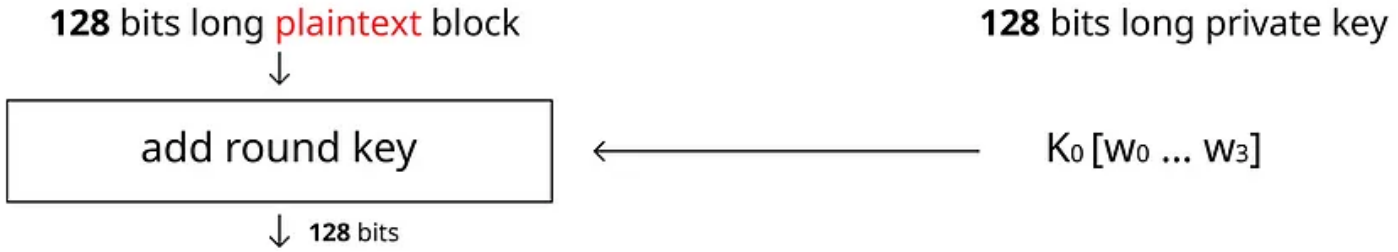
We represent the data (plaintext, ciphertext and key) as metrixes

p_0	p_4	p_8	p_{12}
p_1	p_5	p_9	p_{13}
p_2	p_6	p_{10}	p_{14}
p_3	p_7	p_{11}	p_{15}

k_0	k_4	k_8	k_{12}
k_1	k_5	k_9	k_{13}
k_2	k_6	k_{10}	k_{14}
k_3	k_7	k_{11}	k_{15}

AES CIPHER

- **Add round key**



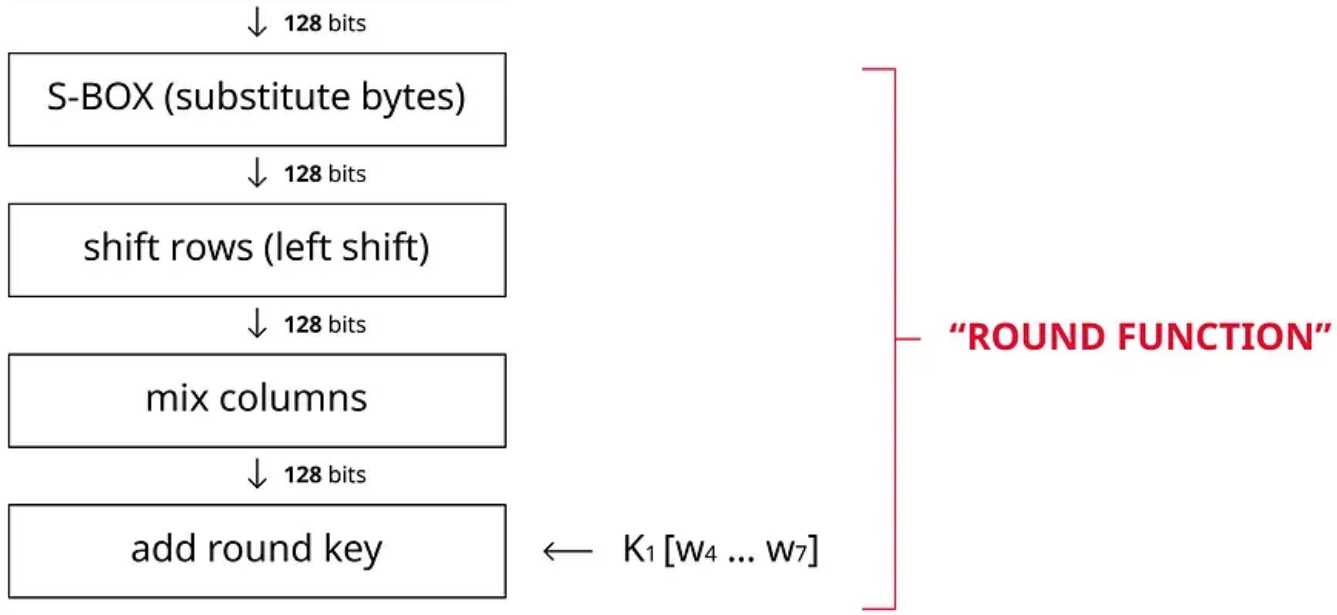
- **Add Round Key** is simply XOR operation.
- We have ***128-bit length*** **Plaintext** and ***128-bit length*** **Round Key** so **XOR operate bit by bit**.
- And as you can see the diagram the probability of having 0 or 1 is 50% each.

output is **0** or **1**
with **50%** probability

x	y	x XOR y
0	0	0
0	1	1
1	0	1
1	1	0

AES CIPHER

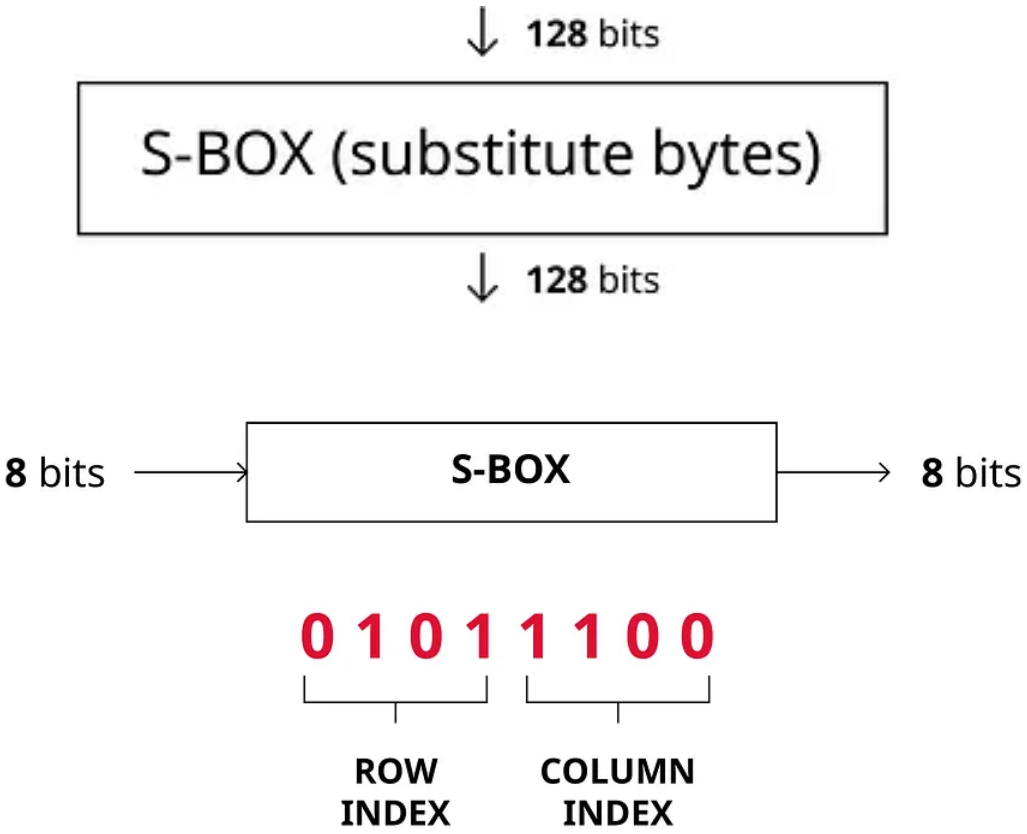
- **Round function**



- We can see the red text "**ROUND FUNCTION**" in the flow chart of AES, which grouped several functions.
- **Round** is simply group of functions, algorithm.
- We can say **executing 10 rounds** as executing 10 times of grouped algorithm.
- ***128-bit length key***, **AES takes 10 rounds**, ***192-bit key*** for **12 rounds** and ***256-bit key*** for **14 rounds**.

AES CIPHER

- **Round function**
  - **Substitute bytes**



AES S-Box. The column is determined by the least significant nibble, and the row by the most significant nibble. For example, the value `0x9a` is converted into `0xb8`.

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

AES CIPHER

- **Round function**
  - **Substitute bytes**

- In the Substitute bytes step, we use S-BOX to substitute data.
- Simply put we can see S-BOX as lookup table.
- The way to substitute bytes for block is like this: each block have 8-bit data, and we can see first 4-bit as row index and the last 4-bit as column index, with these row, column index we can take the value from the S-BOX.

AES S-Box. The column is determined by the least significant nibble, and the row by the most significant nibble. For example, the value `0x9a` is converted into `0xb8`.

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

AES CIPHER

- **Round function**
  - **Substitute bytes**
  - **Shift rows**

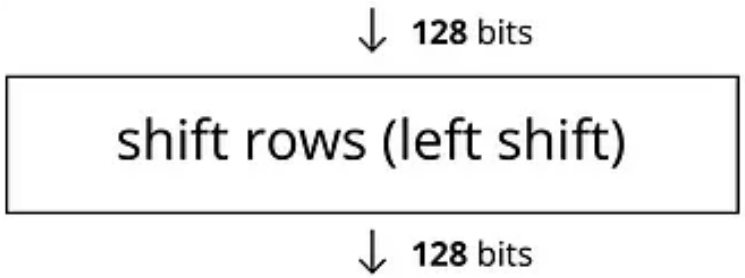
S_0	S_4	S_8	S_{12}
S_1	S_5	S_9	S_{13}
S_2	S_6	S_{10}	S_{14}
S_3	S_7	S_{11}	S_{15}

⟵ circular left shift with 0 step

⟵ circular left shift with 1 steps

⟵ circular left shift with 2 steps

⟵ circular left shift with 3 steps



S_0	S_4	S_8	S_{12}
S_1	S_5	S_9	S_{13}
S_2	S_6	S_{10}	S_{14}
S_3	S_7	S_{11}	S_{15}

⟶

S_0	S_4	S_8	S_{12}
S_5	S_9	S_{13}	S_1
S_{10}	S_{14}	S_2	S_6
S	S_3	S_7	S_{11}

AES CIPHER

- **Round function**
  - **Shift rows**

- In the shift rows section, execute circular left shifting for each row. For first row of box shift 0 step to left, second row of box shift 1 step to left, and so on.
- After finishing shifting rows, first rows changes from s_0, s_4, s_8, s_12 to s_0, s_4, s_8, s_12, second rows changes from s_1, s_5, s_9, s_13 to s_5, s_9, s_13, s_1…

s_0	s_4	s_8	s_{12}
s_1	s_5	s_9	s_{13}
s_2	s_6	s_{10}	s_{14}
s_3	s_7	s_{11}	s_{15}

⟵ circular left shift with 0 step

⟵ circular left shift with 1 steps

⟵ circular left shift with 2 steps

⟵ circular left shift with 3 steps

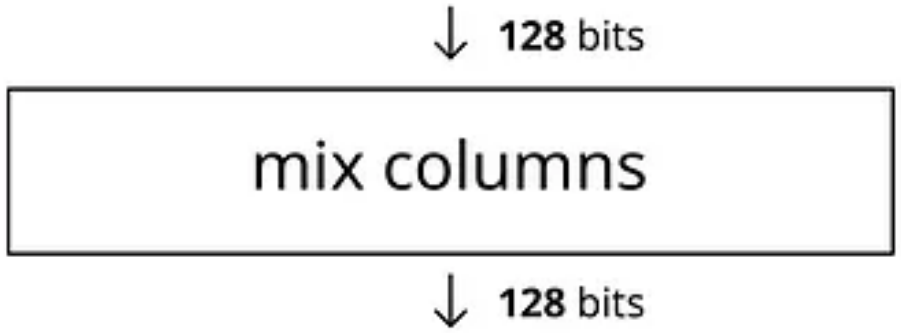
s_0	s_4	s_8	s_{12}
s_1	s_5	s_9	s_{13}
s_2	s_6	s_{10}	s_{14}
s_3	s_7	s_{11}	s_{15}

⟶

s_0	s_4	s_8	s_{12}
s_5	s_9	s_{13}	s_1
s_{10}	s_{14}	s_2	s_6
s	s_3	s_7	s_{11}

AES CIPHER

- **Round function**
  - **…**
  - **Mix columns**



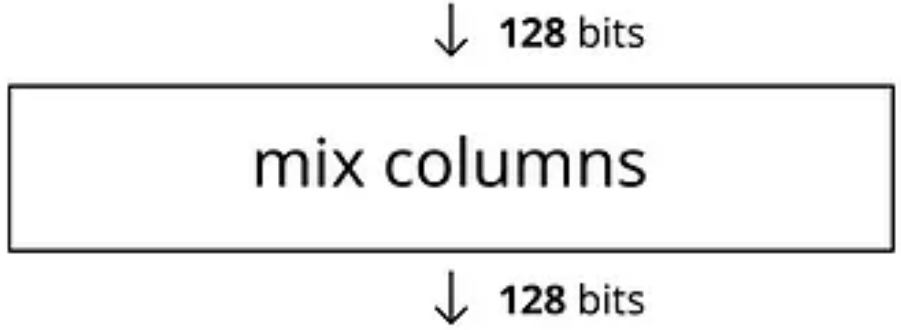
S_0	S_4	S_8	S_{12}
S_1	S_5	S_9	S_{13}
S_2	S_6	S_{10}	S_{14}
S_3	S_7	S_{11}	S_{15}

2	3	1	1
1	2	3	1
1	1	2	3
3	1	1	2

In the mix columns step, execute matrix-vector multiplication column by column. Take one column then multiply it to predefined circulant MDS matrix.

AES CIPHER

- **Round function**
  - **…**
  - **Mix columns**



s_0	s_4	s_8	s_{12}
s_1	s_5	s_9	s_{13}
s_2	s_6	s_{10}	s_{14}
s_3	s_7	s_{11}	s_{15}

→

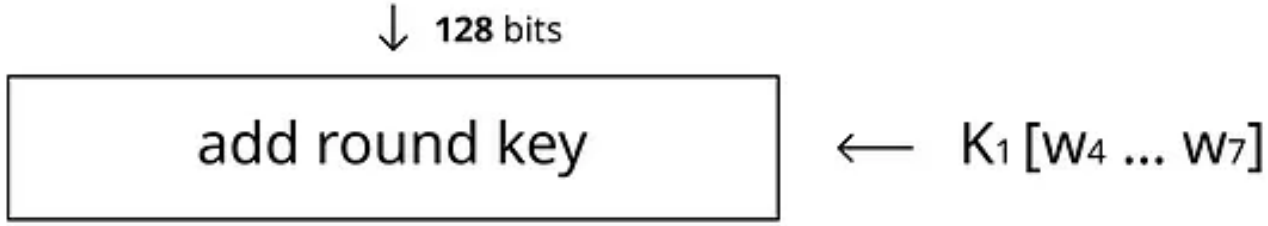
s'_0	s'_4	s'_8	s'_{12}
s'_1	s'_5	s'_9	s'_{13}
s'_2	s'_6	s'_{10}	s'_{14}
s'_3	s'_7	s'_{11}	s'_{15}

After multiplication we do finish mix columns step.

One thing to keep in mind is that mix columns step is not executed in last round.

AES CIPHER

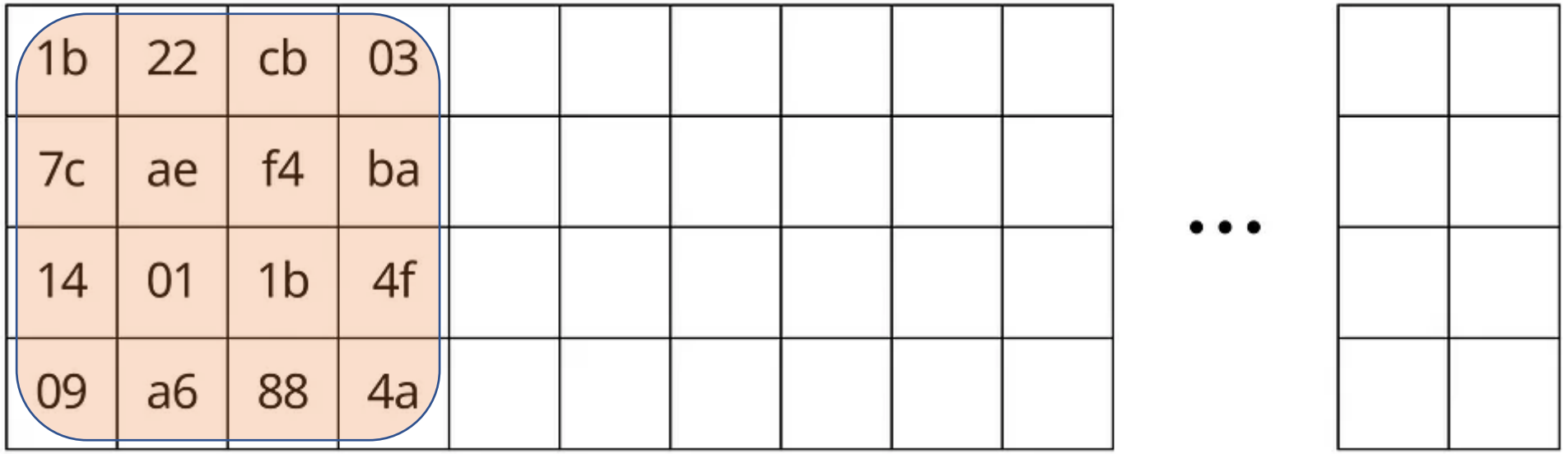
- **Round function**
  - **…**
  - **Add round key**



- And the last step of the round is adding round key.
- At the very first of adding round key step, even before we entered into round, we use our own private key to execute step.
- But in each round, we do not use private key instead we generate subkey and use it to add round key.

AES CIPHER

- **subkey generation**



- We have private key represented as two-dimensional array, and each block has 1 Byte.

AES CIPHER

- subkey generation

1b	22	cb	03								
7c	ae	f4	ba								
14	01	1b	4f								
09	a6	88	4a								

ba
4f
4a
03

- First take the right-most column, and execute circular upward shift

AES CIPHER

- subkey generation

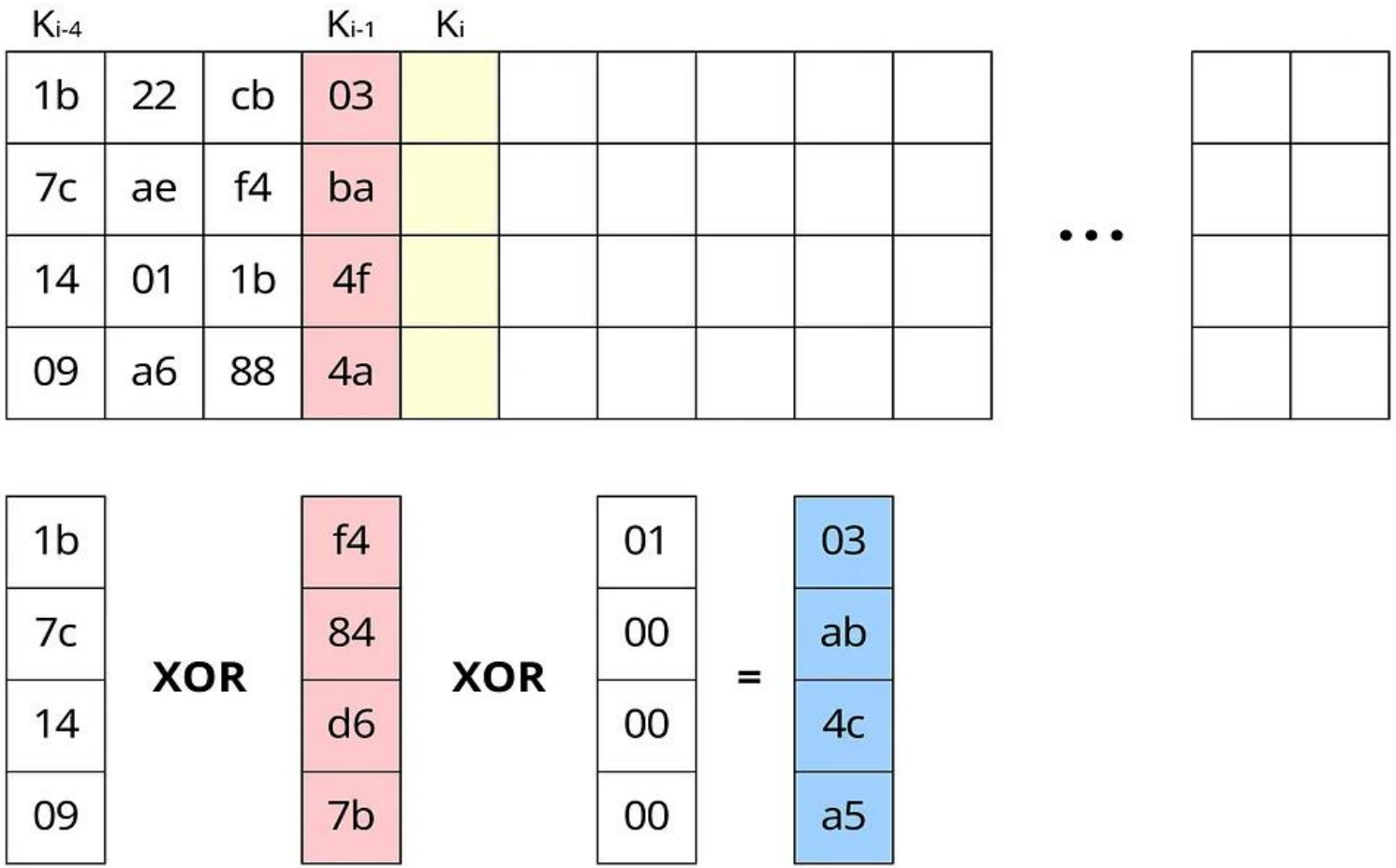
1b	22	cb	03							...	
7c	ae	f4	ba								
14	01	1b	4f								
09	a6	88	4a								

f4
84
d6
7b

- In the same way as we did before in substitute bytes step, substitute bytes using S-BOX

AES CIPHER

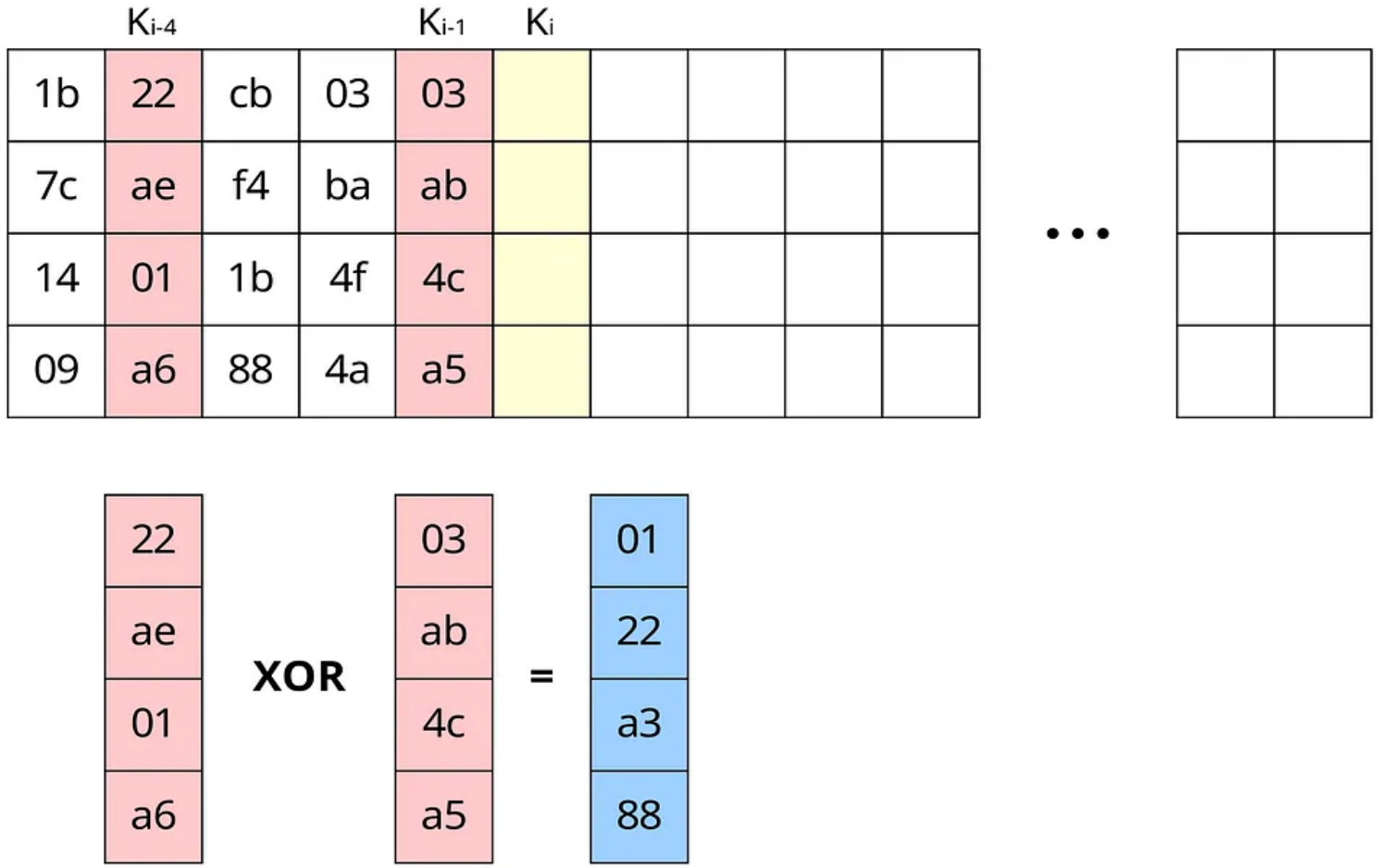
- subkey generation



- Then do XOR operation with K_(i-4) columns and take the predefined value from rcon table and do XOR operation again.
- The result is our first column of current round subkey.

AES CIPHER

- **subkey generation**



- Generating 2nd, 3rd and last column of subkey is rather simple, just do XOR operation on K_(i-1) and K_(i-4) column.

AES CIPHER

- subkey generation

1b	22	cb	03	03	01	f1	23		
7c	ae	f4	ba	ab	22	ac	a3		
14	01	1b	4f	4c	03	02	39		
09	a6	88	4a	a5	88	22	39		

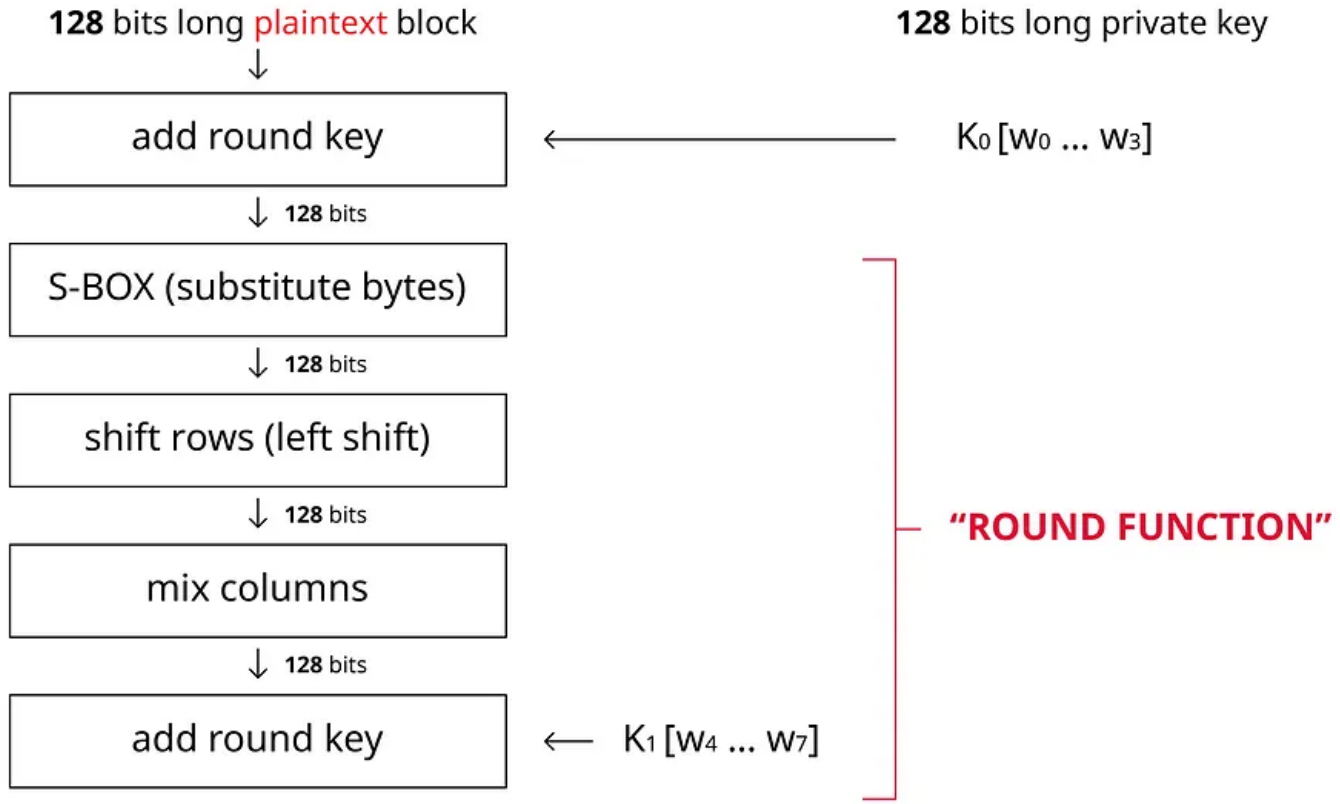
• • •

- Generating 2nd, 3rd and last column of subkey is rather simple, just do XOR operation on K_(i-1) and K_(i-4) column.

AES CIPHER

- **Represent data as matrixes**
- **Add round key**
- **Round function**
  - **Substitute bytes**
  - **Shift rows**
  - **Mix columns**
  - **Add round key**

- **subkey generation**

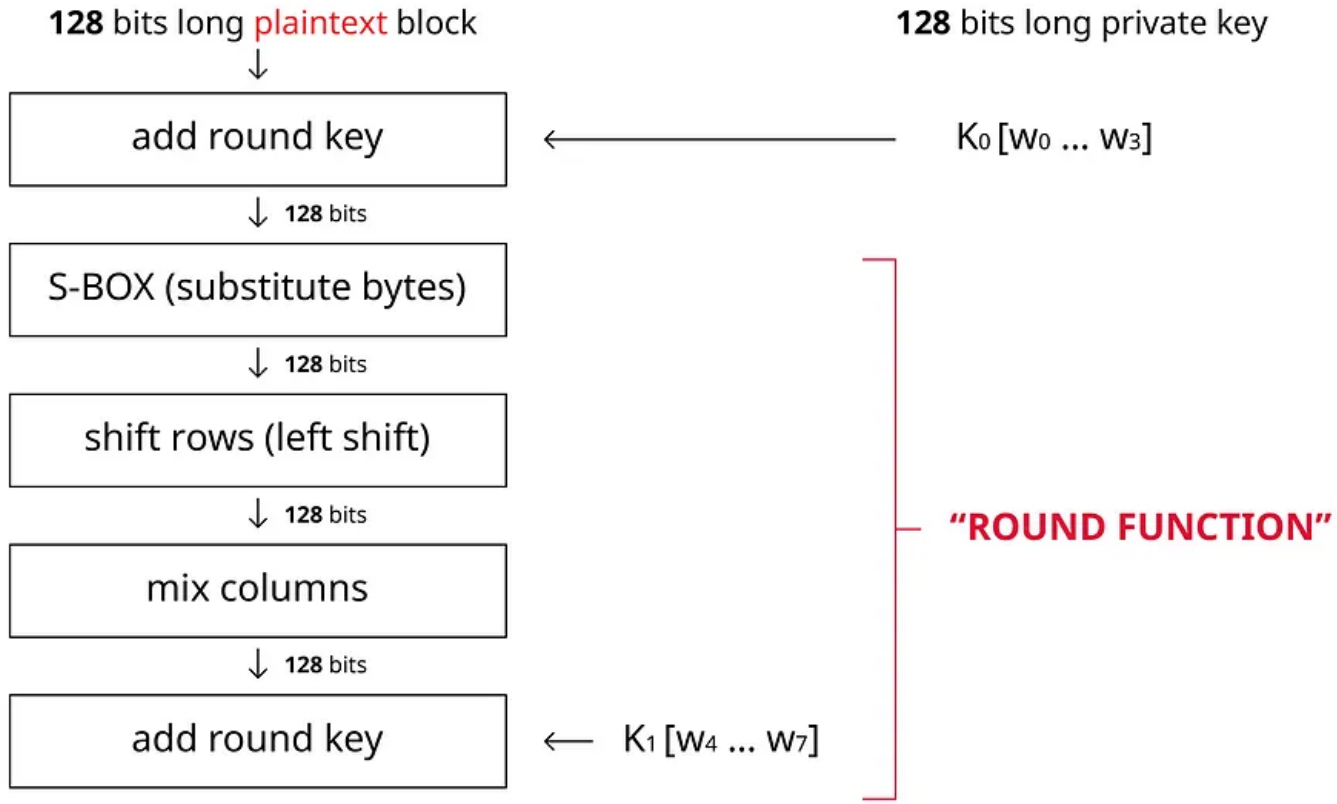


- ✓ We can generate subkey for adding round key in this round, then we do XOR operation with this new subkey and the data we encrypted so far.
- ✓ These are steps AES algorithm takes for each round. And after doing same things for X rounds (10 rounds for 128-bit key length, 12 rounds for 192-bit key length, 14 rounds for 256-bit key length), we can get ciphertext encrypted by AES algorithm.

AES CIPHER

- Represent data as matrixes
- Add round key
- Round function
  - Substitute bytes
  - Shift rows
  - Mix columns
  - Add round key

- subkey generation



AES Example - Input (128 bit key and message)

Key in English: Thats my Kung Fu (16 ASCII characters, 1 byte each)

Plaintext in English: Two One Nine Two (16 ASCII characters, 1 byte each)

AES CIPHER

AES Example - Input (128 bit key and message)

Key in English: Thats my Kung Fu (16 ASCII characters, 1 byte each)

Translation into Hex:

T	h	a	t	s		m	y		K	u	n	g		F	u
54	68	61	74	73	20	6D	79	20	4B	75	6E	67	20	46	75

Key in Hex (128 bits): 54 68 61 74 73 20 6D 79 20 4B 75 6E 67 20 46 75

Plaintext in English: Two One Nine Two (16 ASCII characters, 1 byte each)

Translation into Hex:

T	w	o		O	n	e		N	i	n	e		T	w	o
54	77	6F	20	4F	6E	65	20	4E	69	6E	65	20	54	77	6F

Plaintext in Hex (128 bits): 54 77 6F 20 4F 6E 65 20 4E 69 6E 65 20 54 77 6F

Char	ASCII Code (Decimal)
a	97
b	98
c	99
d	100
e	101
f	102
g	103
h	104
i	105
j	106
k	107
l	108
m	109
n	110
o	111
p	112
q	113
r	114
s	115
t	116
u	117
v	118
w	119
x	120
y	121
z	122

Char	ASCII Code (Decimal)
A	65
B	66
C	67
D	68
E	69
F	70
G	71
H	72
I	73
J	74
K	75
L	76
M	77
N	78
O	79
P	80
Q	81
R	82
S	83
T	84
U	85
V	86
W	87
X	88
Y	89
Z	90

Char	ASCII Code (Decimal)
space	32
!	33
"	34
#	35
$	36
%	37
&	38
'	39
(	40
)	41
*	42
+	43
,	44
-	45
.	46
/	47
:	58
;	59
<	60
=	61
>	62
?	63
@	64
[	91
\	92
]	93
^	94
_	95
`	96
{	123
|	124
}	125
~	126
'	145
'	146
"	147
"	148
•	149
˜	152

Char	ASCII Code (Decimal)
0	48
1	49
2	50
3	51
4	52
5	53
6	54
7	55
8	56
9	57

Char	ASCII Code (Decimal)
€	128
£	163
¥	165
$	36
©	169
™	153
°	176
˜	152
¡	161
¿	191