University of Salahaddin - Erbil
College of Engineering
Department of Software Engineering

Data Security

Academic year 2021-2022
4th Year Material
Chapter Four

Advanced Encryption Standard (AES)

Prepared By: Mr. **Zana Farhad Doghramachi, M.Tech(CSE)**
Zana.softeng@gmail.com

Advanced Encryption Standard

- The Advanced Encryption Standard (AES) is a symmetric-key block cipher published by the National Institute of Standards and Technology (NIST) in December 2000.

- The criteria defined by NIST for selecting AES fall into three areas:

i. **Security**: The main emphasis was on security, NIST explicitly demanded a 128 bit key, this criterion focused on resistance to cryptanalysis attacks other than brute force attack.
ii. **Cost**: which covers the computational efficiency and storage requirement for different implementations such as hardware, software, or smart card.
iii. **Implementation**: the algorithm must have flexibility and simplicity.

Rounds

- AES is a non-Feistel cipher that encrypts and decrypts a data block of 128 bits. It uses 10, 12, or 14 rounds. The key size, which can be 128, 192, or 256 bits, depends on the number of rounds.

- The encryption algorithm and decryption algorithm are similar but the round keys are applied in the reverse order.

- The figure in the next page, shows the relationship between the number of rounds and the key size, which means that we can have three different AES versions; they are referred as AES-128, AES-192, and AES-256. However, the round keys, which are created by the key-expansion algorithm are always 128 bits, the same size as the plaintext or ciphertext block.

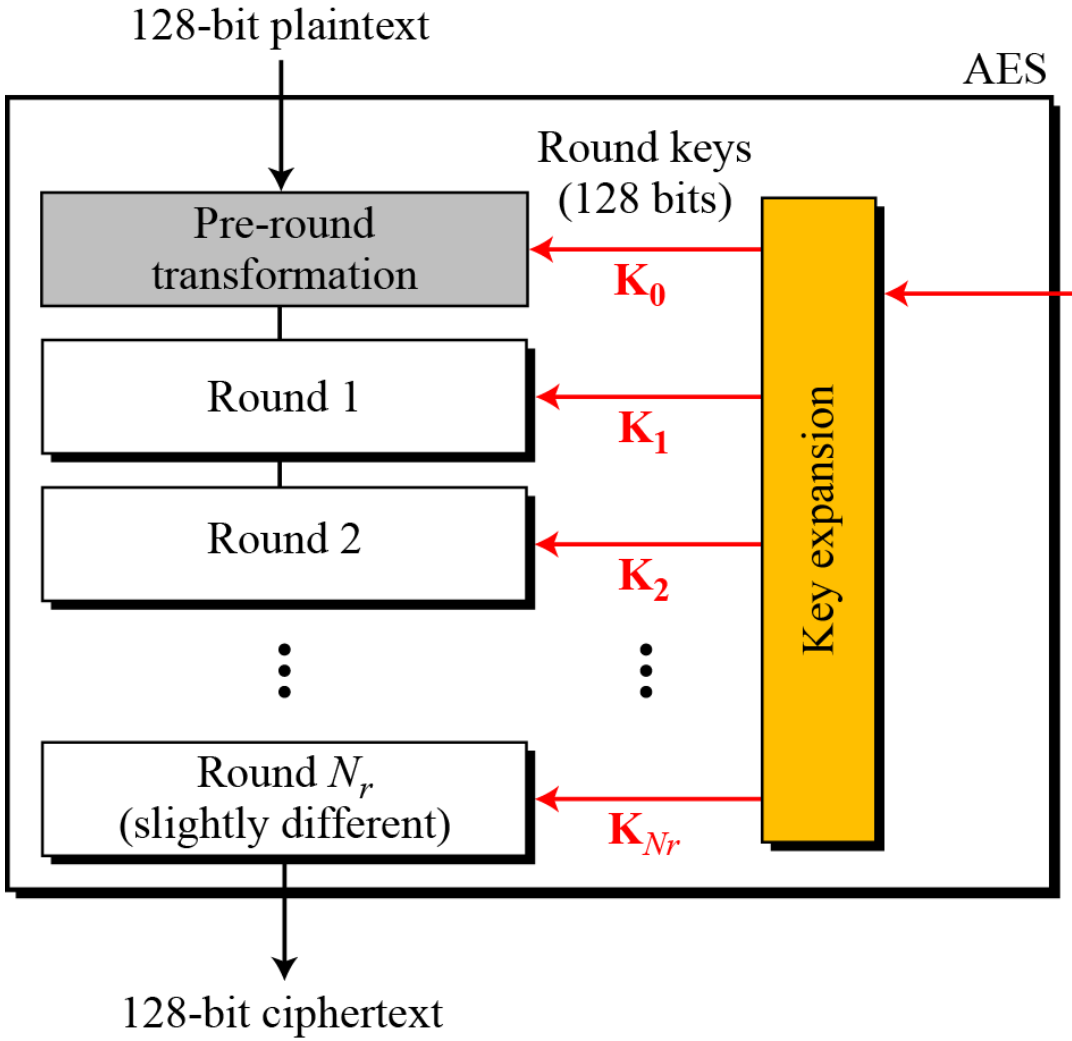
Rounds

- The number of round keys generated by the key expansion algorithm is always one more than the number of round. In other words, we have

  number of round keys = number of round (Nr)+ 1

  We refer to the round keys as K_0, K_1, K_2,……..,K_{Nr} .

General Design of AES



Cipher key
(128, 192, or 256 bits)

Nr	Key size
10	128
12	192
14	256

Relationship between number of rounds and cipher key size

Data Units

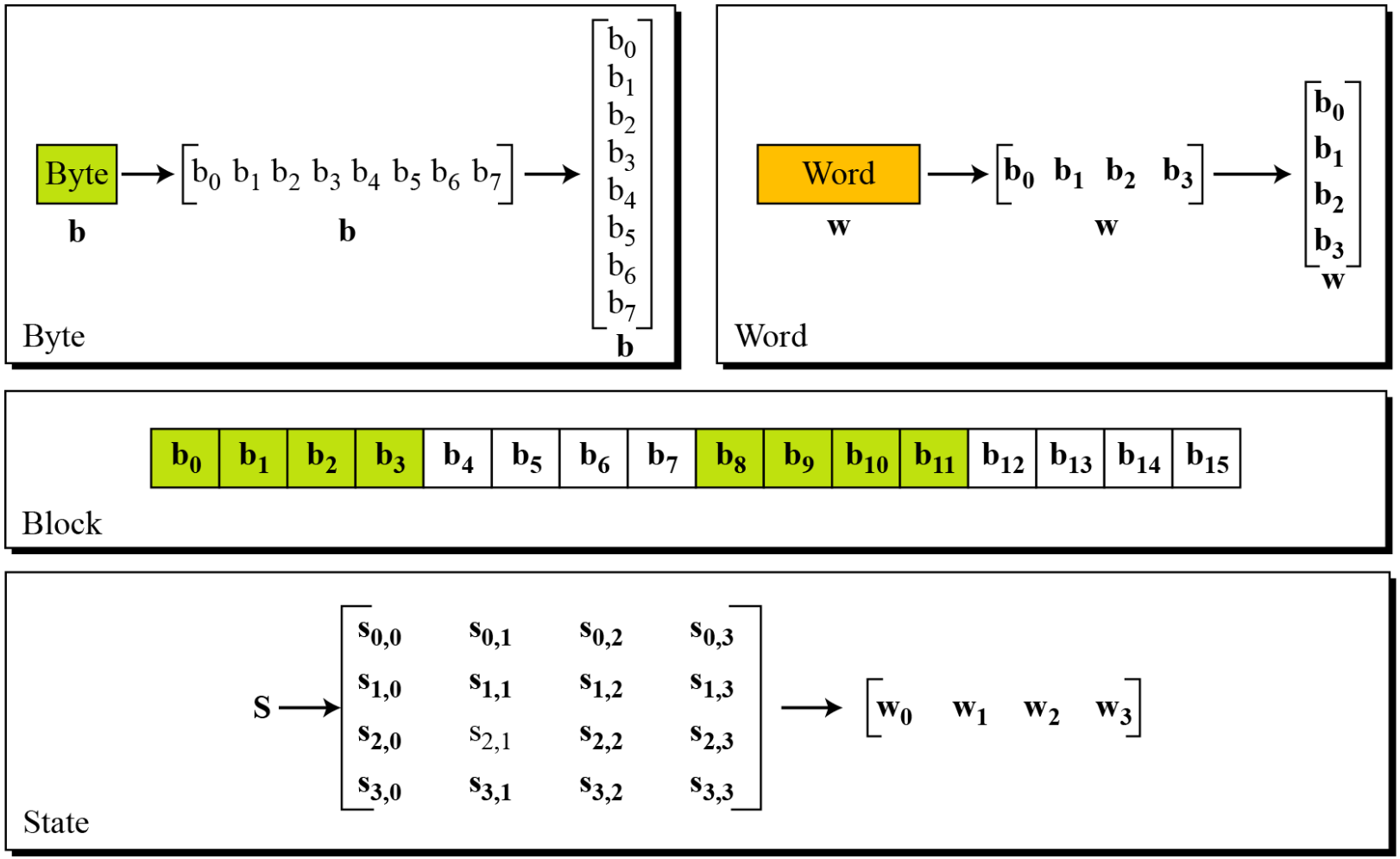
- AES uses five units of measurement to refer to data: bits, bytes, words, blocks, and state. The bit is the smallest and atomic unit.
- A **bit** is a binary digit with a value 0 or 1.
- A **byte** is a group of eight bits that can be treated as a single entity. A row matrix (1_x8) of eight bits, or a column matrix (8_x1) of eight bits. When treated as a row matrix, the bits are inserted to the matrix from left to right; when treated as a column matrix, the bits are inserted into the matrix from top to bottom.
- A **word** is a group of 32 bits that can be treated as a single entity, a row matrix of four bytes, or a column of four bytes. When treated as a row matrix, the bits are inserted to the matrix from left to right; when treated as a column matrix, the bits are inserted into the matrix from top to bottom.

Data Units

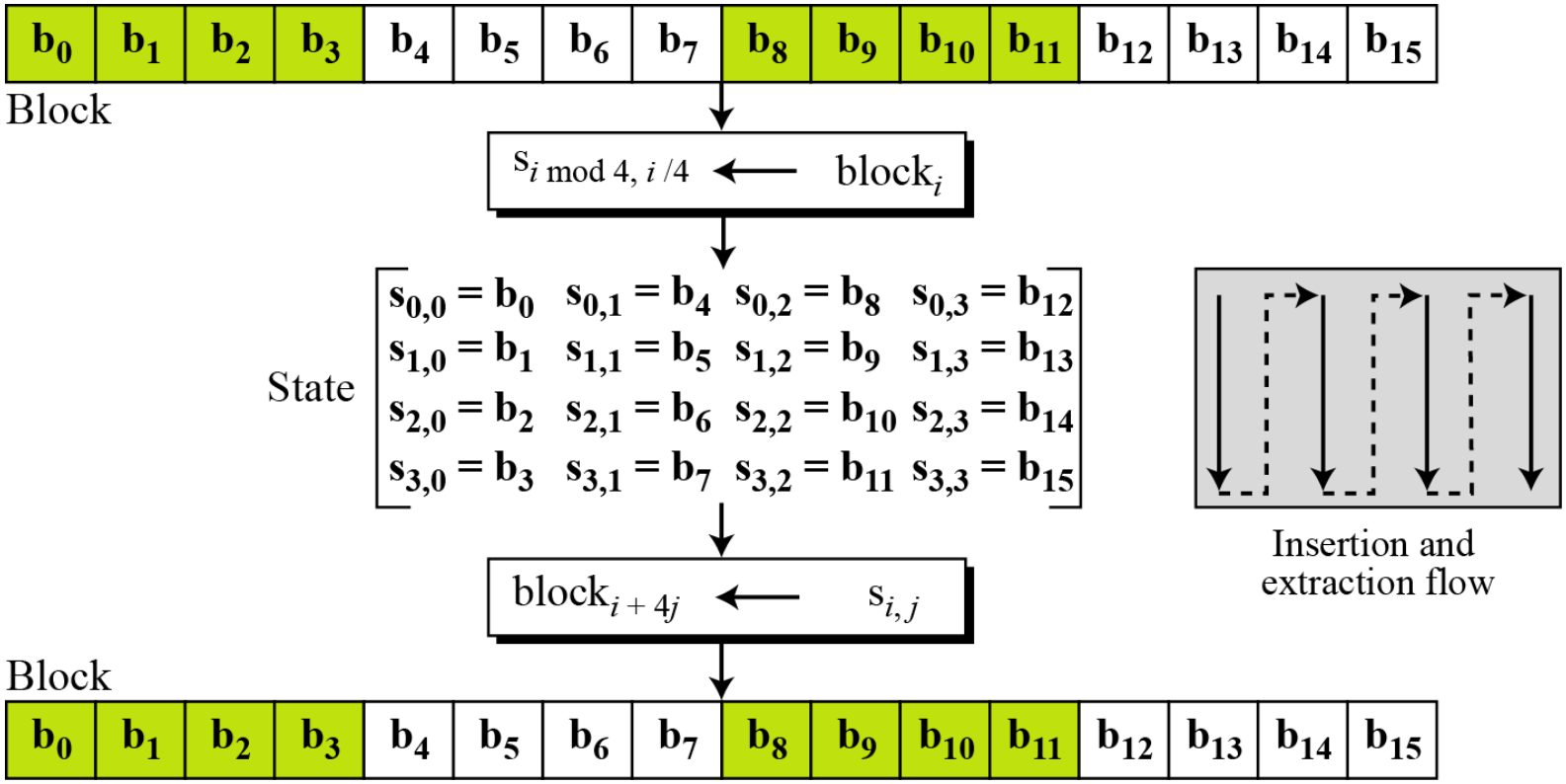
- A **block** in AES is a group of 128 bits. However, a block can be represented as a row matrix of 16 bytes.

- **State** is made of 16 bytes, but normally is treated as matrices of 4_x4 bytes. Each element of a state is referred to as $s_{r,c}$, where r (0 to 3) defines the row and the c (0 to 3) defines the column. Occasionally, a state is treated as a row matrix (1_x4) of words

Data units used in AES



Block to state & state to block transformation



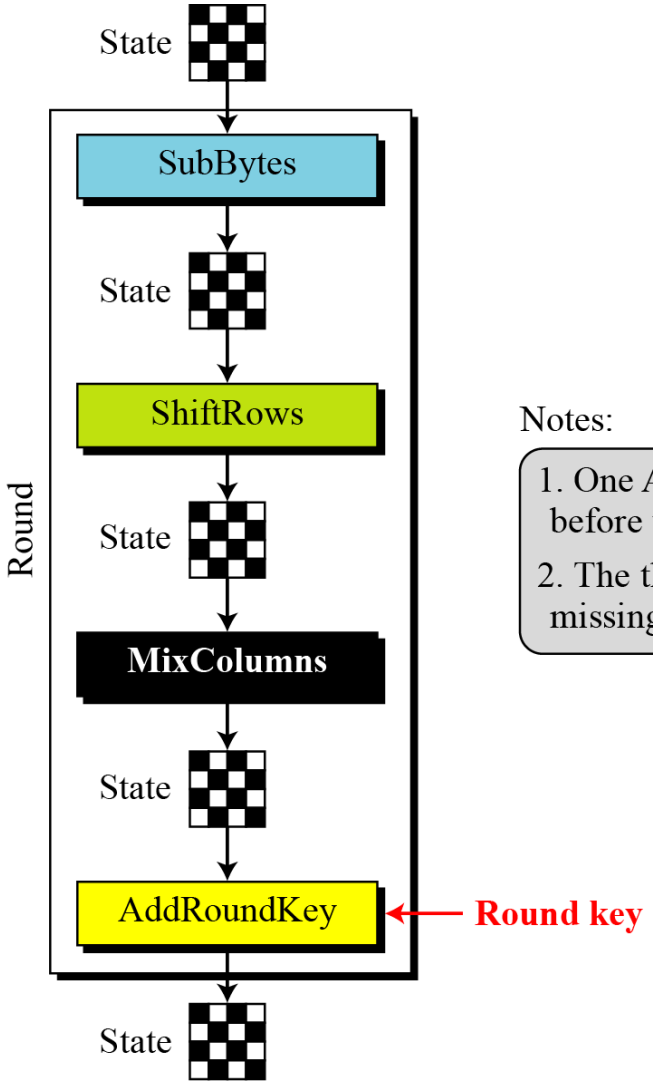
Homework

- Q1 Change the plaintext to state, if you're using AES encryption and the plaintext is "AES uses a matrix".

Structure of Each Round

- Each round, except the last, uses four transformations that are invertible. The last round has only three transformations.
- AES transformations are: substitution(Sub Bytes), permutation(Shift Rows), mixing (Mix Columns), and key adding (Add Round Key).
- Each transformation takes a state and creates another state to be used for the next transformation or the next round. The pre-round section uses only one transformation (Add round Key); the last round uses only three transformation (Mix columns transformation is missing).
- At the decryption site, the inverse transformations are used.

Structure of each round at the encryption site



Notes:

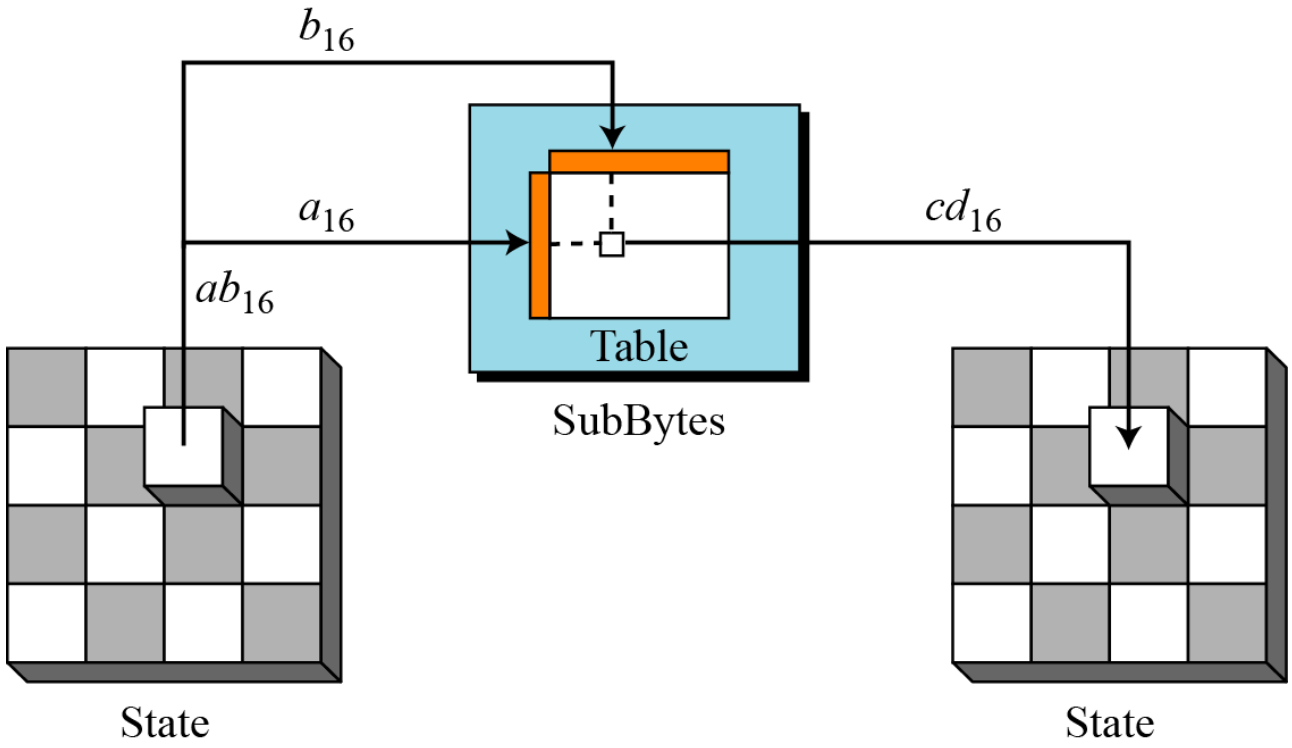
1. One AddRoundKey is applied before the first round.
2. The third transformation is missing in the last round.

Sub Bytes

- To substitute a byte, we interpret the byte as two hexadecimal digits. The left digit defines the row and the right digit defines the column of the substitution table. The two hexadecimal digits at the junction of the row and the column are the new byte.

- In the SubBytes transformation, the state is treated as a 4_x4 matrix of bytes. Transformation is done one byte at a time. The contents of each byte is changed, but the arrangement of the bytes in the matrix remains the same. In the process, each byte is transformed independently. There are sixteen distinct byte to byte transformation.

Sub Bytes transformations



Sub Byte transformation table

	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	CB	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

InvSub Byte transformation table

	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

Homework

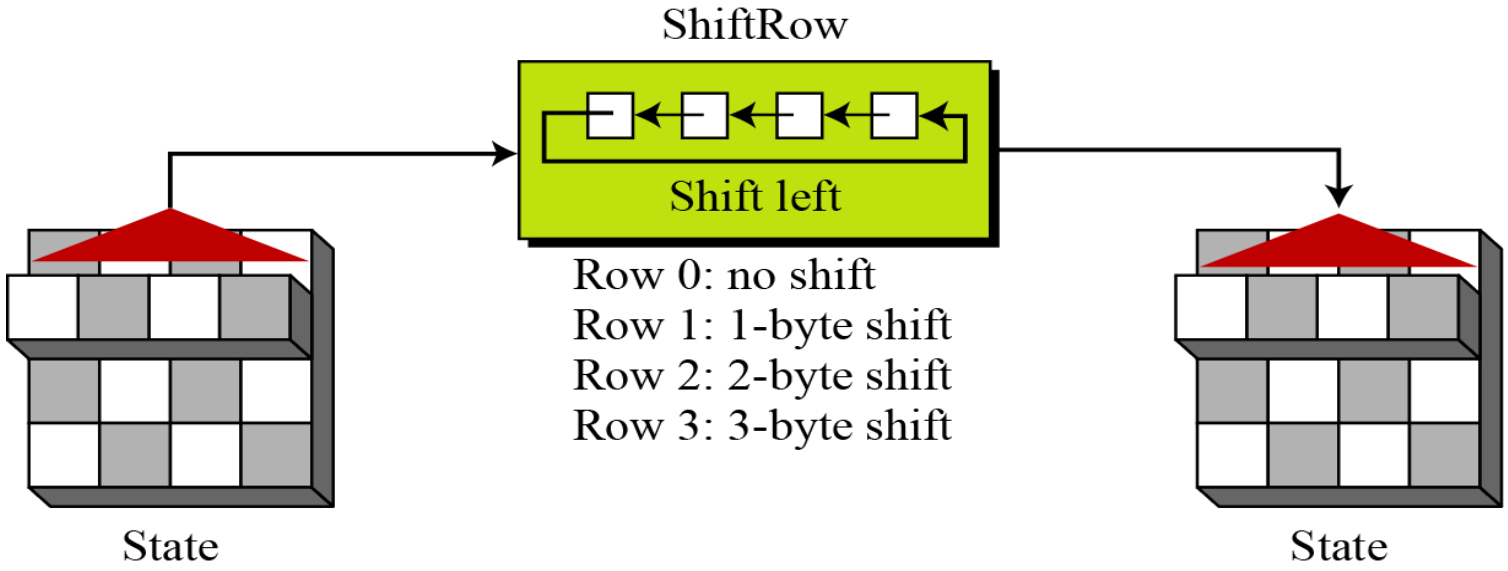
- Q1 Transform the state bye using the Sub Bytes transformation?

$$\text{State} \begin{bmatrix} 00 & 12 & 0C & 08 \\ 04 & 04 & 00 & 23 \\ 12 & 12 & 13 & 19 \\ 14 & 00 & 11 & 19 \end{bmatrix}$$

Shift Rows

- In the encryption, the shifting is to the left and the number of shifts depends on the row number (0, 1, 2 or 3) of the state matrix. This means the row 0 is not shifted at all and the last row is shifted three bytes.

- In the decryption, the shifting is to the right. The number is the same the row number (0, 1, 2, and 3) of the state matrix

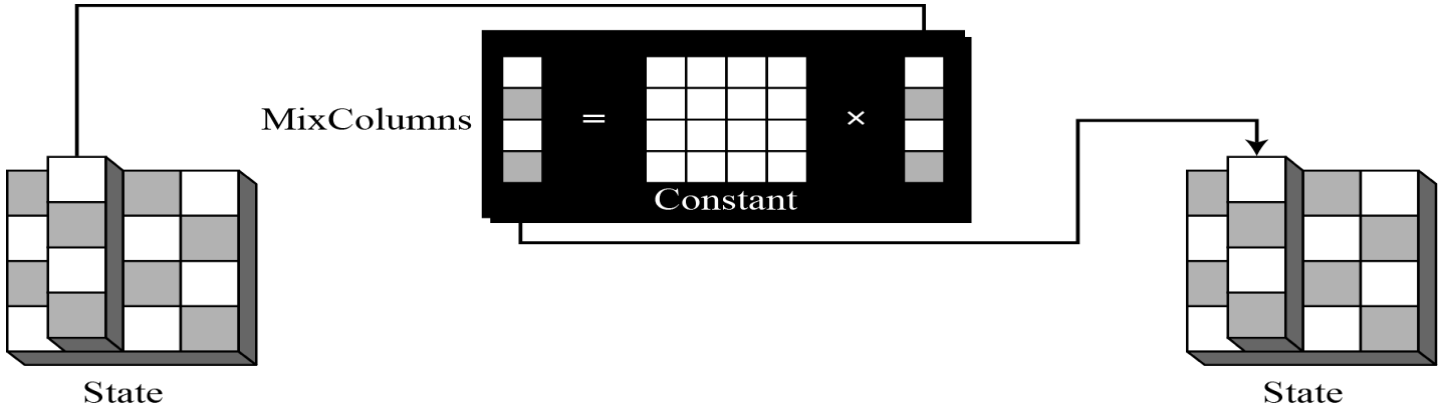


Mix Columns

- The mixing transformation changes the contents of each byte by taking four bytes at a time and combining them to recreate four new bytes. To guarantee that each new byte is different (even if all four bytes are the same), the combination process first multiples each byte with a different constant and then mixes them.

- The mixing can be provided by matrix multiplication. When we multiply a square matrix by a column matrix, the result is a new column. Each element in the new matrix depends on all four elements of the old matrix after they are multiplied by row values in the constant matrix

Mixing bytes using matrix multiplication

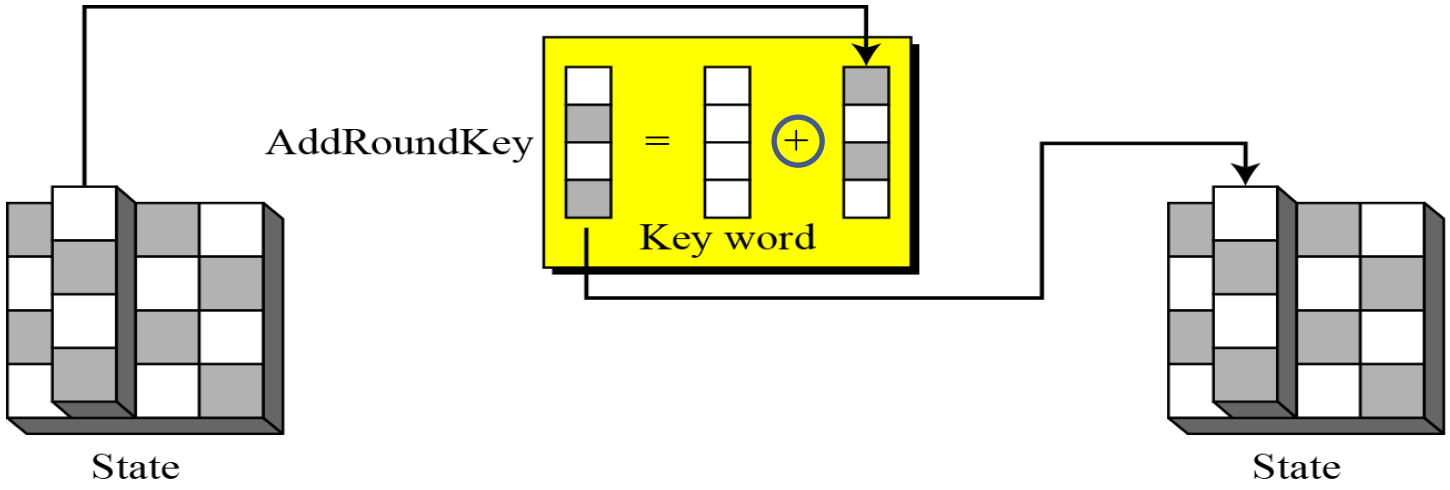


$$\begin{bmatrix} ax + by + cz + dt \\ ex + fy + gz + ht \\ ix + jy + kz + lt \\ mx + ny + oz + pt \end{bmatrix} = \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{bmatrix} \times \begin{bmatrix} x \\ y \\ z \\ t \end{bmatrix}$$

New matrix — **Constant matrix** — Old matrix

Add Round Key

- Proceeds one column at a time, Add Round Key adds a round key word with each state column matrix, so the operation in add round key is matrix addition.



Key Expansion

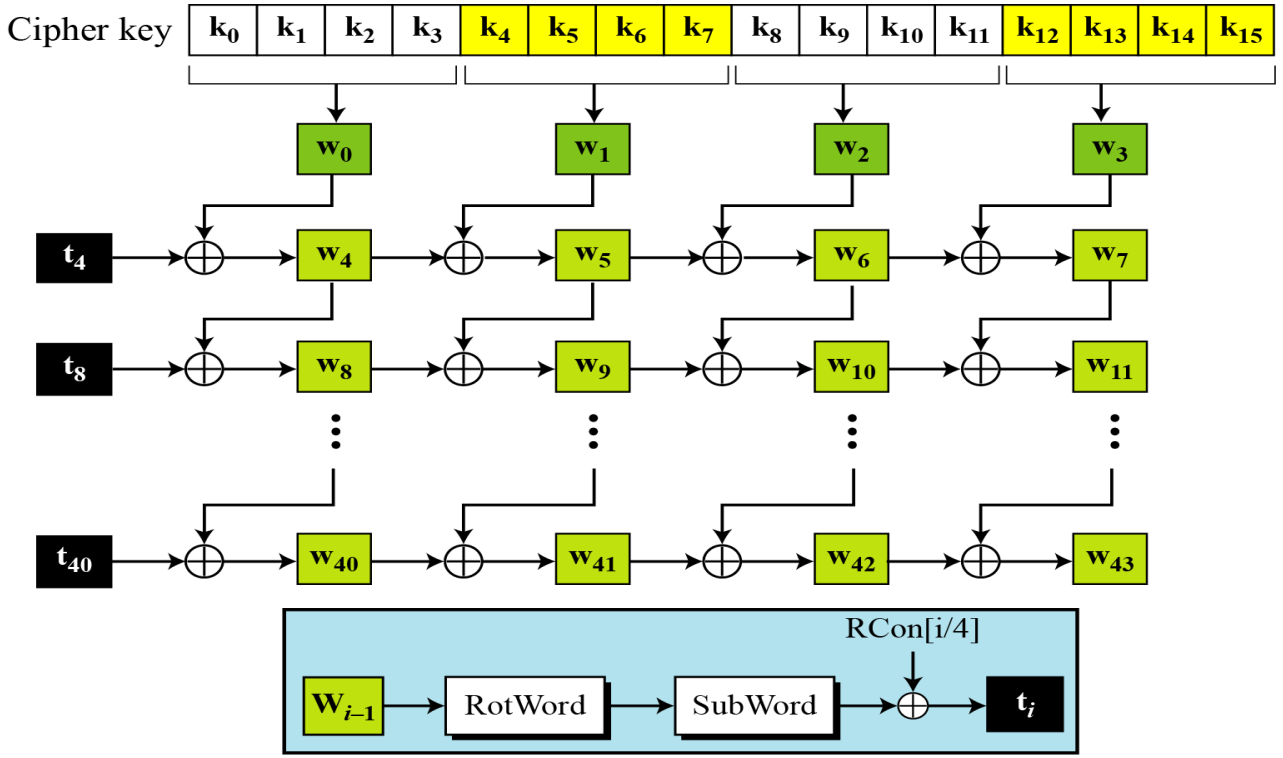
- To create round keys for each round, AES uses a key expansion process. If the number of rounds is N_r , the key-expansion routine creates N_r+1 128 bit round keys from one single 128-bit cipher key. The first round key is used for pre-round transformation; the remaining round keys are used for the last transformation at the end of each round.

- The key expansion routine creates round keys word by word, where a word is an array for four bytes. The routine creates 4_x (N_r+1) words that are called $w_0, w_1, w_2, \ldots\ldots, w_{4(Nr+1)-1}$

Round	*Words*			
Pre-round	$\mathbf{w}_0$	$\mathbf{w}_1$	$\mathbf{w}_2$	$\mathbf{w}_3$
1	$\mathbf{w}_4$	$\mathbf{w}_5$	$\mathbf{w}_6$	$\mathbf{w}_7$
2	$\mathbf{w}_8$	$\mathbf{w}_9$	$\mathbf{w}_{10}$	$\mathbf{w}_{11}$
. . .	. . .			
N_r	$\mathbf{w}_{4N_r}$	$\mathbf{w}_{4N_r+1}$	$\mathbf{w}_{4N_r+2}$	$\mathbf{w}_{4N_r+3}$

Key Expansion

- The 44 words key are made from the original key, the processes for the other two versions are the same with some slight changes.



Making of t_i (temporary) words $i = 4\ N_r$

Key Expansion

The process is as follows:

1. The first four words (w_0, w_1, w_2, w_3) are made from the cipher key. The cipher key is thought of as an array of 16 bytes (k_0 to k_{15}). The first four bytes (k_0 to k_3) become w_0; the next four bytes (k_4 to k_7) become w_1; and son on.

2. The rest of words (w_i for i=4 to 43) are made as follow:

a) If (i mod 4) $\neq$ 0, $w_i = w_{i-1} \oplus w_{i-4}$. This means each word is made from the one at the left and the one at the top.

b) If (i mod 4)=0, $w_i = t \oplus w_{i-4}$. Here t, a temporary word, is the result of applying two routines SubWord and RotWord on w_{i-1} and XORing the result with a round constant, Rcon. In other words we have,

$t = \text{SubWord}(\text{RotWord}(w_{i-1})) \oplus \text{Rcon}_{i/4}$

Key Expansion

- The **RotWord** (rotate word) routine is similar to the shift rows transformation, but it is applied to only one row. The routine takes a word as an array of four bytes and shifts each byte to the left with wrapping.

- The **SubWord** (substitute word) routine is similar to the Sub Bytes transformation, but it is applied only to four bytes, The routine takes each byte in the word and substitutes another byte for it.

- Each round constant, RCon, is a 4 byte value in which the rightmost three bytes are always zero.

Rcon constants AES-128

Round	*Constant (RCon)*	*Round*	*Constant (RCon)*
1	$(\mathbf{01}\ 00\ 00\ 00)_{16}$	6	$(\mathbf{20}\ 00\ 00\ 00)_{16}$
2	$(\mathbf{02}\ 00\ 00\ 00)_{16}$	7	$(\mathbf{40}\ 00\ 00\ 00)_{16}$
3	$(\mathbf{04}\ 00\ 00\ 00)_{16}$	8	$(\mathbf{80}\ 00\ 00\ 00)_{16}$
4	$(\mathbf{08}\ 00\ 00\ 00)_{16}$	9	$(\mathbf{1B}\ 00\ 00\ 00)_{16}$
5	$(\mathbf{10}\ 00\ 00\ 00)_{16}$	10	$(\mathbf{36}\ 00\ 00\ 00)_{16}$

Homework

- Q1 Explain key expansion in AES 192 and AES 256?

Analysis of AES

AES was designed after DES. Most of the known attacks on DES were already tested on AES.

- **Brute force attacks**

  AES is definitely more secure than DES due to the larger size key (128, 192 and 256 bits), we need 2^{128} tests to find the key.

- **Statistical attacks**

  Numerous tests have failed to do statistical analysis of the ciphertext. Because strong diffusion and confusion provided by the combination of the Sub Byte, Shift Rows and Mix Columns transformations remove any frequency pattern in the plaintext.

- **Differential and Linear attacks**

  There are no differential and linear attacks on AES as yet.