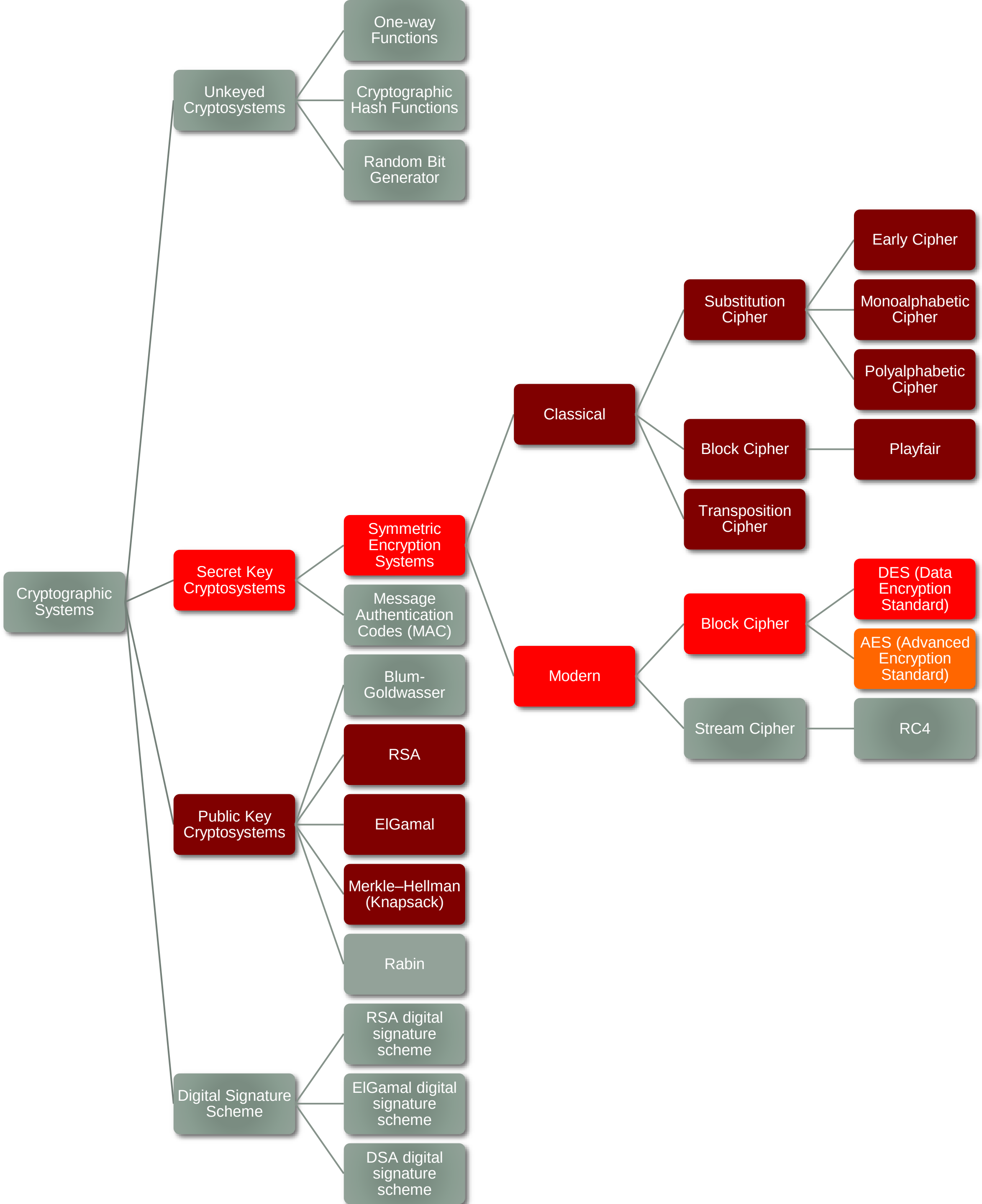
TUTORIAL

CSCI361 – Computer Security

Sionggo Japit

sjapit@uow.edu.au

12 February 2024


Cryptographic Systems
Unkeyed Cryptosystems
One-way Functions
Cryptographic Hash Functions
Random Bit Generator
Secret Key Cryptosystems
Symmetric Encryption Systems
Message Authentication Codes (MAC)
Classical
Substitution Cipher
Early Cipher
Monoalphabetic Cipher
Polyalphabetic Cipher
Block Cipher
Playfair
Transposition Cipher
Modern
Block Cipher
DES (Data Encryption Standard)
AES (Advanced Encryption Standard)
Stream Cipher
RC4
Public Key Cryptosystems
Blum-Goldwasser
RSA
ElGamal
Merkle–Hellman (Knapsack)
Rabin
Digital Signature Scheme
RSA digital signature scheme
ElGamal digital signature scheme
DSA digital signature scheme

AES (ADVANCED ENCRYPTION STANDARD)

AES

- AES is a block cipher with a block length of 128 bits and a variable key length of 128, 192, or 256 bits.
- The number of rounds depends on the key length (i.e., 10, 12, or 14 rounds).

	Block Length	Key Length	Number of rounds
AES-128	4	4	10
AES-192	4	6	12
AES-256	4	8	14

AES

- AES operates on a two-dimensional array s of bytes, called the *State.*
- The State consists of 4 rows and N_b columns (where N_b is the block length divided by 32).
- In the current AES specification, N_b is always 4 (for all official versions of the AES). Note, however, that this need not be the case and that there may be future versions of the AES that work with larger values for N_b.

AES

- Each entry in the State refers to a byte $s_{r,c}$ or s[r, c] (where $0 \leq r < 4$ refers to the row number and $0 \leq c < 4$ refers to the column number).
- With $N_b = 4$ (current specification), each state has 16 bytes.

AES

Encryption:

- At the start of the encryption process, the 16 input bytes are copied into the state s.
- An initial application of the add-round-key transformation is applied to the state s.
- Next 10, 12, or 14 round-function are applied to the state s, with a final round that slightly differs from the previous $N_r - 1$ rounds (i.e., the final round does not include a MixColumns() transformation).
- The content of the State is finally taken to represent the output of the AES encryption algorithm.

AES

AES encryption algorithm:

(in)

$s \leftarrow in$
$s \leftarrow \text{AddRoundKey}(s, w[0, N_b - 1])$
for r=1 to $(N_r - 1)$ do
    $s \leftarrow \text{SubBytes}(s)$
    $s \leftarrow \text{ShiftRows}(s)$
    $s \leftarrow \text{MixColumns}(s)$
    $s \leftarrow \text{AddRoundKey}(s, w[rN_b, (r + 1)N_b - 1])$
$s \leftarrow \text{SubBytes}(s)$
$s \leftarrow \text{ShiftRows}(s)$
$s \leftarrow \text{AddRoundKey}(s, w[N_r N_b , (N_r + 1)N_b - 1])$
$out \leftarrow s$

(out)

AES

Decryption:

- The transformations used by the AES encryption algorithm can be inverted and implemented in reverse order to produce a straightforward AES decryption algorithm.
- The individual transformations used in the AES decryption algorithm are called InvShiftRows(), InvSubBytes(), InvMixColumns(), and AddRoundKey().
- Since the add-round-key transformation involves a bitwise addition modulo 2, the add-round-key function is its own inverse. Also, note that the SubBytes() and ShiftRows() transformations commute, and that this is also true for their inverse InvSubBytes() and InvShiftRows() transformations.

AES

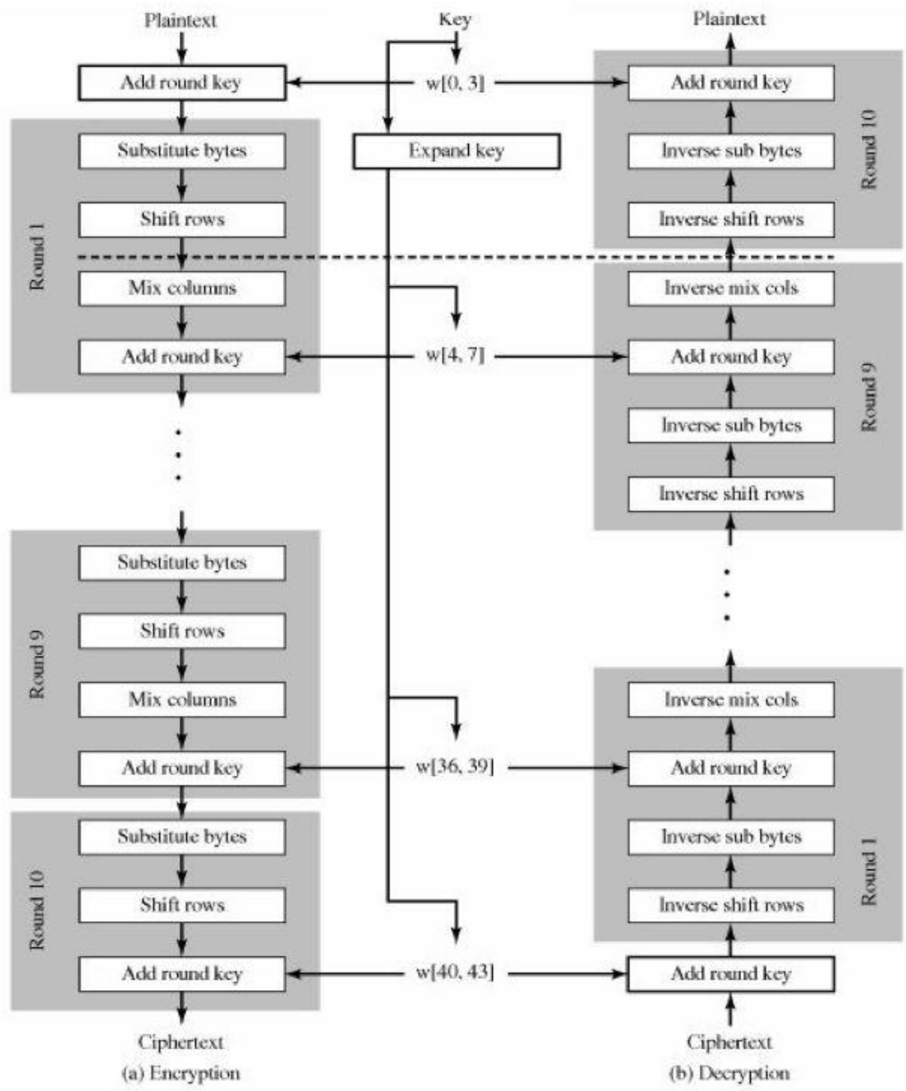
AES decryption algorithm:

(in)

$s \leftarrow in$
$s \leftarrow \text{AddRoundKey}(s, w[N_r N_b\ , (N_r + 1)N_b - 1])$
for $r = N_r - 1$ downto 1 do
    $s \leftarrow \text{InvShiftRows}(s)$
    $s \leftarrow \text{InvSubBytes}(s)$
    $s \leftarrow \text{AddRoundKey}(s, w[rN_b, (r + 1)N_b - 1])$
    $s \leftarrow \text{InvMixColumns}(s)$
$s \leftarrow \text{InvShiftRows}(s)$
$s \leftarrow \text{InvSubBytes}(s)$
$s \leftarrow \text{AddRoundKey}(s, w[0, N_b - 1])$
$out \leftarrow s$ (out)

(out)

AES



Plaintext
Key
Plaintext
Add round key
w[0, 3]
Add round key
Substitute bytes
Expand key
Inverse sub bytes
Round 10
Shift rows
Inverse shift rows
Round 1
Mix columns
Inverse mix cols
Add round key
w[4, 7]
Add round key
Inverse sub bytes
Round 9
Inverse shift rows
Substitute bytes
Shift rows
Round 9
Mix columns
Inverse mix cols
Add round key
w[36, 39]
Add round key
Substitute bytes
Inverse sub bytes
Round 10
Shift rows
Inverse shift rows
Round 1
Add round key
w[40, 43]
Add round key
Ciphertext
Ciphertext
(a) Encryption
(b) Decryption

AES Stages

The four AES's invertible operations:

1. Byte substitution (S-box, 8-bit to 8-bit).
2. Shift row (rotating order of bytes in each row).
3. Mix column (linear mixing of a word column).
4. Key mixing (addition).

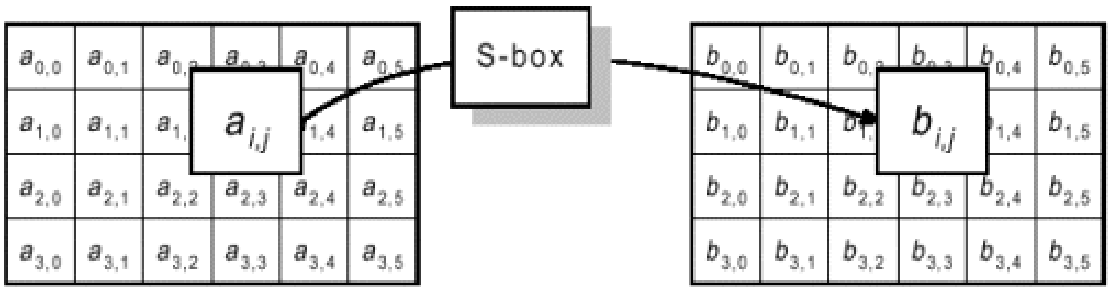
BYTE SUBSTITUTION (SUB-BYTE TRANSFORMATION)

Substitute Bytes Transformation

- The byte substitution is a fixed S box from 8 bits (**x**) to 8 bits (**y**) or 16 x 16 matrix.
- It substitutes all the bytes of the input according to the following matrix:

```
y=S[x] = {
        99, 124, 119, 123, 242, 107, 111, 197,  48,   1, 103,  43, 254, 215, 171, 118,
        202, 130, 201, 125, 250,  89,  71, 240, 173, 212, 162, 175, 156, 164, 114,
192,
        183, 253, 147,  38,  54,  63, 247, 204,  52, 165, 229, 241, 113, 216,  49,  21,
        4, 199,  35, 195,  24, 150,   5, 154,   7,  18, 128, 226, 235,  39, 178, 117,
        9, 131,  44,  26,  27, 110,  90, 160,  82,  59, 214, 179,  41, 227,  47, 132,
        83, 209,   0, 237,  32, 252, 177,  91, 106, 203, 190,  57,  74,  76,  88, 207,
        208, 239, 170, 251,  67,  77,  51, 133,  69, 249,   2, 127,  80,  60, 159, 168,
        81, 163,  64, 143, 146, 157,  56, 245, 188, 182, 218,  33,  16, 255, 243, 210,
        205,  12,  19, 236,  95, 151,  68,  23, 196, 167, 126,  61, 100,  93,  25, 115,
        96, 129,  79, 220,  34,  42, 144, 136,  70, 238, 184,  20, 222,  94,  11, 219,
        224,  50,  58,  10,  73,   6,  36,  92, 194, 211, 172,  98, 145, 149, 228, 121,
        231, 200,  55, 109, 141, 213,  78, 169, 108,  86, 244, 234, 101, 122, 174,   8,
        186, 120,  37,  46,  28, 166, 180, 198, 232, 221, 116,  31,  75, 189, 139, 138,
        112,  62, 181, 102,  72,   3, 246,  14,  97,  53,  87, 185, 134, 193,  29, 158,
        225, 248, 152,  17, 105, 217, 142, 148, 155,  30, 135, 233, 206,  85,  40, 223,
        140, 161, 137,  13, 191, 230,  66, 104,  65, 153,  45,  15, 176,  84, 187,  22
        };
```

Substitute Bytes Transformation



Each byte at the input of a round undergoes a
non-linear byte substitution according to the following transform:

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

Substitution (“S”)-box

AES S-box (in Hexadecimal)

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

Substitute Bytes Transformation

Example of the SubBytes transformation

EA	04	65	85
83	45	5D	96
5C	33	98	B0
F0	2D	AD	C5

87	F2	4D	97
EC	6E	4C	90
4A	C3	46	E7
8C	D8	95	A6

The hexadecimal value {65} references row 6, column 5 of the S-box, which contains the value {4D}.

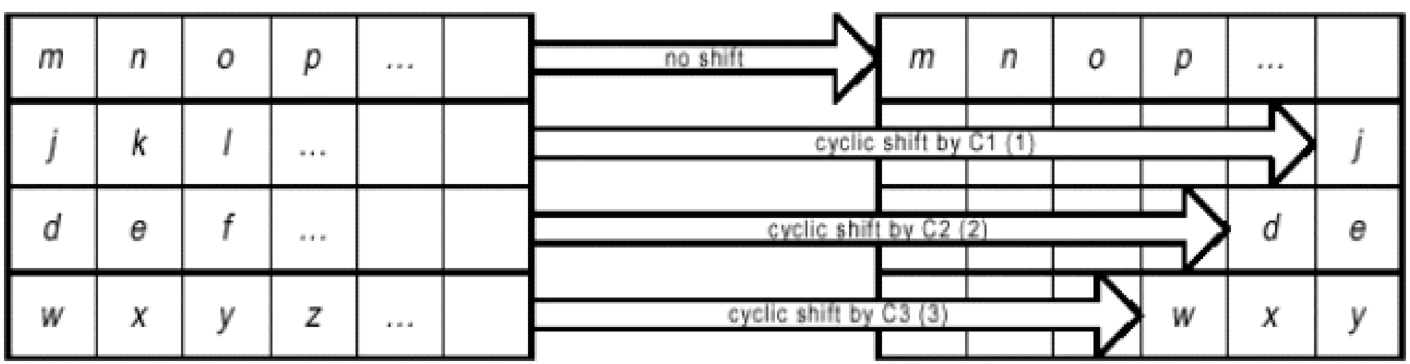
SHIFT ROW TRANSFORMATION

ShiftRows Transformation

- The Shift row operation rotates the order of bytes.

Nb	C1	C2	C3
4	1	2	3
6	1	2	3
8	1	3	4

Depending on the block length, each "row" of the block is cyclically shifted according to the above table



ShiftRows Transformation

Example of ShiftRows transformation

EA	04	65	85
83	45	5D	96
5C	33	98	B0
F0	2D	AD	C5

→

EA	04	65	85
45	5D	96	83
98	B0	5C	33
C5	F0	2D	AD

- The first row of State is not altered.
- The second row is left-rotate by 1-byte.
- The third row is left-rotate by 2-byte.
- The fourth row is left-rotate by 3-byte.

MIXCOLUMNS TRANSFORMATION

MixColumns Transformation

- Mix column is a linear mixing of a word column (four bytes) ($a_{i,j}$, $a_{i,j}$, $a_{i,j}$, $a_{i,j}$) into a word column ($b_{i,j}$, $b_{i,j}$, $b_{i,j}$, $b_{i,j}$), where i and j = 1, 2, 3, and 4. The mixing is defined by operations in the field GF(2^8), and can be defined by the following matrix multiplication on State.

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \times \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \\ a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} \\ a_{4,1} & a_{4,2} & a_{4,3} & a_{4,4} \end{bmatrix} = \begin{bmatrix} b_{1,1} & b_{1,2} & b_{1,3} & b_{1,4} \\ b_{2,1} & b_{2,2} & b_{2,3} & b_{2,4} \\ b_{3,1} & b_{3,2} & b_{3,3} & b_{3,4} \\ b_{4,1} & b_{4,2} & b_{4,3} & b_{4,4} \end{bmatrix}$$

Coefficient. This is fixed (constant).

State (matrix) before mixcolumn process.

State (matrix) after mixcolumn process.

MixColumns Transformation

- Each element in the product matrix is the sum of products of elements of one row and one column performed in GF(2^8), with the irreducible polynomial $m(x) = x^8 + x^4 + x^3 + x + 1$ (also known as **prime polynomial**).
- The MixColumns transformation on a single column j ($0 \le j \le 3$) of **State** can be expressed as:

$$S'_{0,j} = (2 \cdot S_{0,j}) \oplus (3 \cdot S_{1,j}) \oplus S_{2,j} \oplus S_{3,j}$$

$$S'_{1,j} = S_{0,j} \oplus (2 \cdot S_{1,j}) \oplus (3 \cdot S_{2,j}) \oplus S_{3,j}$$

$$S'_{2,j} = S_{0,j} \oplus S_{1,j} \oplus (2 \cdot S_{2,j}) \oplus (3 \cdot S_{3,j})$$

$$S'_{3,j} = (3 \cdot S_{0,j}) \oplus S_{1,j} \oplus S_{2,j} \oplus (2 \cdot S_{3,j})$$

MixColumns Transformation

- An example of MixColumns:

D4	F2	4D	97
BF	4C	90	EC
5D	E7	4A	C3
30	8C	D8	95

→

04	40	A3	4C
66	D4	70	9F
81	E4	3A	42
E5	A5	A6	BC

MixColumns Transformation

- Note: in $GF(2^8)$,
  - Addition is the bitwise XOR operation, and
  - multiplication is performed according to the following rule:
    - Multiplication of a value by x, where x = 02, can be implemented as a 1-bit left shift followed by a conditional bitwise XOR with (0001 1011) if the leftmost bit of the original value (prior to the shift) is 1.

(This is because AES uses arithmetic in the finite field $GF(2^8)$, with the irreducible polynomial $m(x) = x^8 + x^4 + x^3 + x + 1$, and thus $(100011011)_{bin} = (11b)_{hex} = (283)_{dec}$.)

MODULAR ARITHMETIC VS MODULAR POLYNOMIAL ARITHMETIC

Modular arithmetic vs modular polynomial arithmetic

Before we look at the example on mixcolumn, we have a short revision on modular arithmetic.

For example, we want to perform a **two-digit integer** with a **three-digit modulo**. Let the modulo be a prime number 107, we can then do the calculation as follow:

- 70 mod 107 = 70
- 68 mod 107 = 68

Now if we double the integer by multiplying it with 2, we have 2 x 70 = 140 mod 107 = 33. But since we can only perform a two-digit integer modulus with three-digit modulo, how can we do 140 mod 107?

Modular arithmetic vs modular polynomial arithmetic

What is 100 mod 107? 100 mod 107 = 100

What is -7 mod 107? -7 mod 107 = 100

Through this observation, we can conclude that 100 mod 107 congruent (100 – 107) mod 107 or 100 mod 107 congruent -7 mod 107.

Now we can say 140 mod 107 is equivalent to 100 mod 107 + 40 mod 107.
We can then substitute 100 mod 107 with -7, and we have
-7 + 40 mod 107 = 33.

Modular arithmetic vs modular polynomial arithmetic

Similarly operation can be done with polynomial in GF(2^8).

$$Let\ m(x) = (x^8 + x^4 + x^3 + x + 1)\ and$$

$$f(x) = b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x^1 + b_0$$

If we multiply *f(x)* by *x*, we have

$$x \times f(x) = (b_8x^8 + b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x^1 + b_0x) \bmod m(x)$$

$$where\ m(x) = (x^8 + x^4 + x^3 + x + 1)$$

Modular arithmetic vs modular polynomial arithmetic

$$Note:(b_8x^8+b_7x^7+b_6x^6+b_5x^5+b_4x^4+b_3x^3+b_2x^2+b_1x^1+b_0x)\bmod m(x)$$

is equivalent to

$$b_8x^8 \bmod (x^8+x^4+x^3+x+1)+$$

$$b_7x^7+b_6x^6+b_5x^5+b_4x^4+b_3x^3+b_2x^2+b_1x^1+b_0x) \bmod (x^8+x^4+x^3+x+1)$$

Through similar observation,

$$x^8 \bmod (x^8+x^4+x^3+x+1)=(x^8+x^4+x^3+x+1)-\ x^8 \bmod (x^8+x^4+x^3+x+1)$$

$$=(x^4+x^3+x+1)$$

Modular arithmetic vs modular polynomial arithmetic

Hence, $(b_8x^8 + b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x^1 + b_0x) \bmod m(x)$

is equivalent to

$$b_8x^8 \bmod (x^8 + x^4 + x^3 + x + 1) +$$

$$b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x^1 + b_0x) \bmod (x^8 + x^4 + x^3 + x + 1)$$

$$= \left(x^8 + x^4 + x^3 + x + 1\right) +$$

$$b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x^1 + b_0x)$$

Modular arithmetic vs modular polynomial arithmetic

Thus we can generalize that

$$x \times f(x) = \begin{cases} (b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x^1 + b_0x) & \text{if } b_7 = 0 \\ (b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x^1 + b_0x) + (x^4 + x^3 + x + 1) & \text{if } b_7 = 1 \end{cases}$$

Note the definition (generalization) is performed on modular polynomial arithmetic.

MIXCOLUMNS TRANSFORMATION

MixColumns Transformation

- For example, the first column transformation of the example shown can be obtained as follows:

$$(D4 \cdot 02) \oplus (BF \cdot 03) \oplus 5D \oplus 30$$

$$(D4 \cdot 02) = (1010\,1000) \oplus (0001\,1011) = (1011\,0011)$$

$$\begin{aligned}(BF \cdot 03) &= (BF) \oplus (BF \cdot 02) \\ &= (1011\,1111) \oplus (0111\,1110 \oplus 0001\,1011) \\ &= (1101\,1010)\end{aligned}$$

MixColumns Transformation

(D4 • 02) : (1011 0011)

(BF • 03) : (1101 1010)

(5D): (0101 1101)

(30): (0011 0000)

(0000 0100) = (04)

MixColumns Transformation

$D4 \oplus (BF \bullet 02) \oplus (5D \bullet 03) \oplus 30$

$(D4) =$ **(1101 0100)**

$(BF \bullet 02) =$ **(0110 0101)**

$(5D \bullet 03) = 5D \oplus (5D \bullet 02)$

$= (0101\ 1101) \oplus (1011\ 1010)$

$=$ **(1110 0111)**

MixColumns Transformation

(30) = **(0011 0000)**

(1101 0100) ⊕ (0110 0101) ⊕ (1110 0111) ⊕ (0011 0000)

(1101 0100)
(0110 0101)
(1110 0111)
<u>(0011 0000)</u>
(0110 0110) = (66)

MixColumns Transformation

$D4 \oplus BF \oplus (5D \bullet 02) \oplus (30 \bullet 03)$

(D4) = **(1101 0100)**

(BF) = **(1011 1111)**

(5D • 02) = **(1011 1010)**

$(30 \bullet 03) = (30) \oplus (30 \bullet 02)$

$= (0011\ 0000) \oplus (0110\ 0000)$

= **(0101 0000)**

MixColumns Transformation

(1101 0100) $\oplus$ (1011 1111) $\oplus$ (1011 1010) $\oplus$ (0101 0000)

(1101 0100)

(1011 1111)

(1011 1010)

<u>(0101 0000)</u>

(1000 0001) = (81)

MixColumns Transformation

$(D4 \bullet 03) \oplus BF \oplus 5D \oplus (30 \bullet 02)$

$(D4 \bullet 03) = D4 \oplus (D4 \bullet 02)$

$= (1101\ 0100) \oplus (1011\ 0011)$

$=$ **(0110 0111)**

BF = **(1011 1111)**

5D = **(0101 1101)**

$(30 \bullet 02) =$ **(0110 0000)**

MixColumns Transformation

(0110 0111) ⊕ (1011 1111) ⊕ (0101 1101) ⊕ (0110 0000)

(0110 0111)

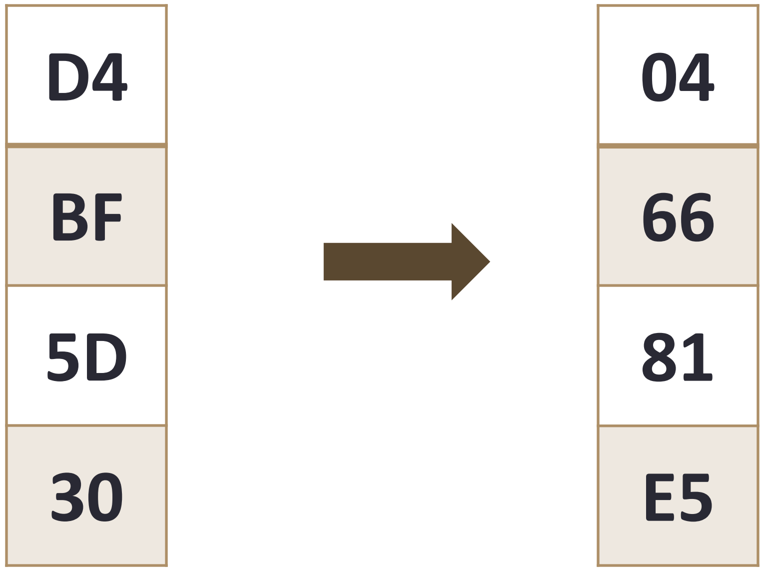
(1011 1111)

(0101 1101)

<u>(0110 0000)</u>

(1110 0101) = (E5)

MixColumns Transformation



KEY MIXING (ADD ROUND KEY)

Key mixing (add round)

- In the key mixing transformation, called Add-round-key, the 128 bits of State are bitwise Xored (added modulo 2) with the 128 bits of the round key.

$$
\begin{bmatrix} S_{0,0} & S_{0,1} & S_{0,2} & S_{0,3} \\ S_{1,0} & S_{1,1} & S_{1,2} & S_{1,3} \\ S_{2,0} & S_{2,1} & S_{2,2} & S_{2,3} \\ S_{3,0} & S_{3,1} & S_{3,2} & S_{3,3} \end{bmatrix}
\oplus
\begin{bmatrix} W_{0,0} & W_{0,1} & W_{0,2} & W_{0,3} \\ W_{1,0} & W_{1,1} & W_{1,2} & W_{1,3} \\ W_{2,0} & W_{2,1} & W_{2,2} & W_{2,3} \\ W_{3,0} & W_{3,1} & W_{3,2} & W_{3,3} \end{bmatrix}
=
\begin{bmatrix} S'_{0,0} & S'_{0,1} & S'_{0,2} & S'_{0,3} \\ S'_{1,0} & S'_{1,1} & S'_{1,2} & S'_{1,3} \\ S'_{2,0} & S'_{2,1} & S'_{2,2} & S'_{2,3} \\ S'_{3,0} & S'_{3,1} & S'_{3,2} & S'_{3,3} \end{bmatrix}
$$

State

Round key

Key mixing (add round)

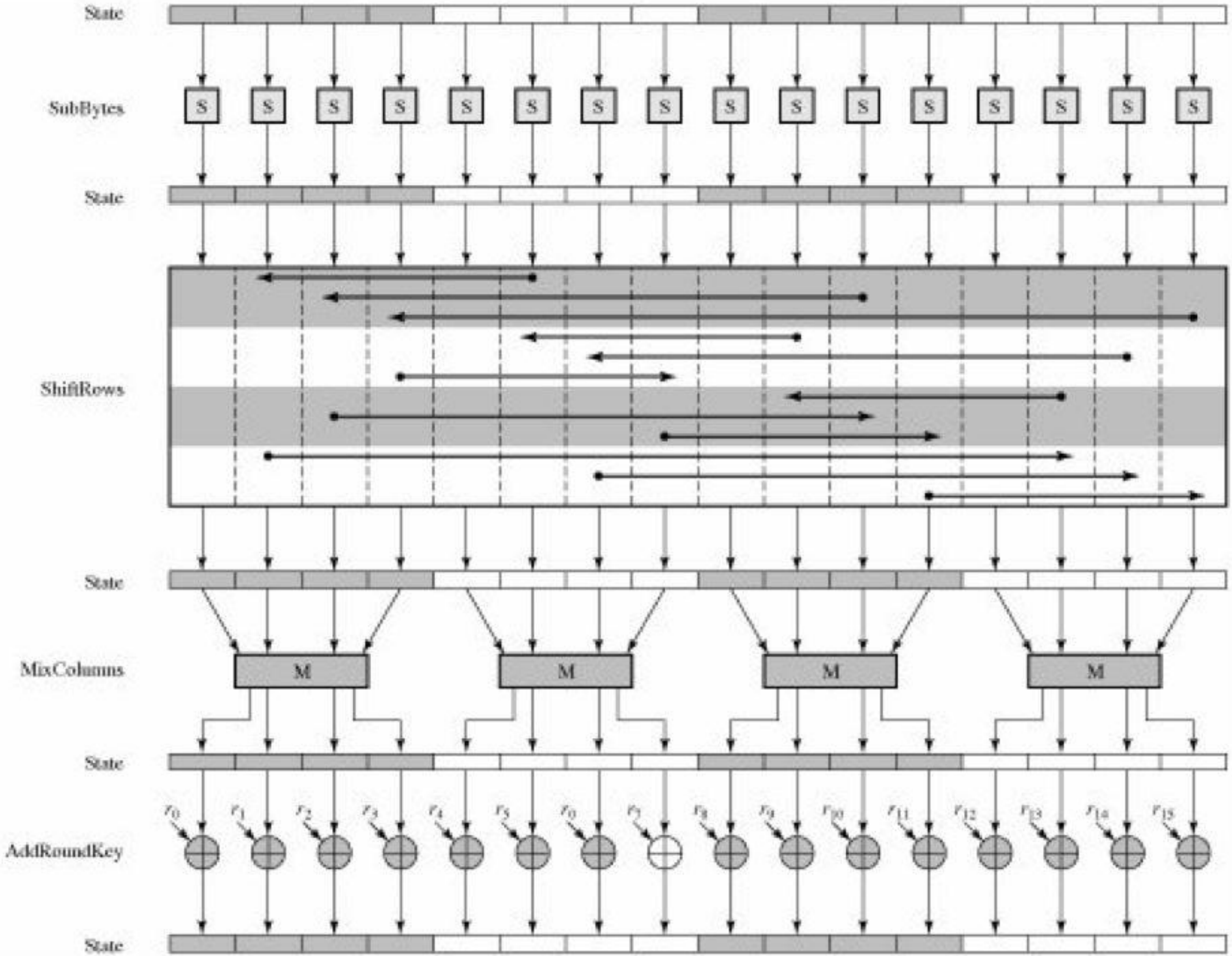
For example:

$$\begin{bmatrix} 04 & 40 & A3 & 4C \\ 66 & D4 & 70 & 9F \\ 81 & E4 & 3A & 42 \\ E5 & A5 & A6 & BC \end{bmatrix} \oplus \begin{bmatrix} AC & 19 & 28 & 57 \\ 77 & FA & D1 & 5C \\ 66 & DC & 29 & 00 \\ F3 & 21 & 41 & 6A \end{bmatrix} = \begin{bmatrix} A8 & 59 & 8B & 1B \\ 11 & 2E & A1 & C3 \\ E7 & 38 & 13 & 42 \\ 16 & 84 & E7 & D2 \end{bmatrix}$$

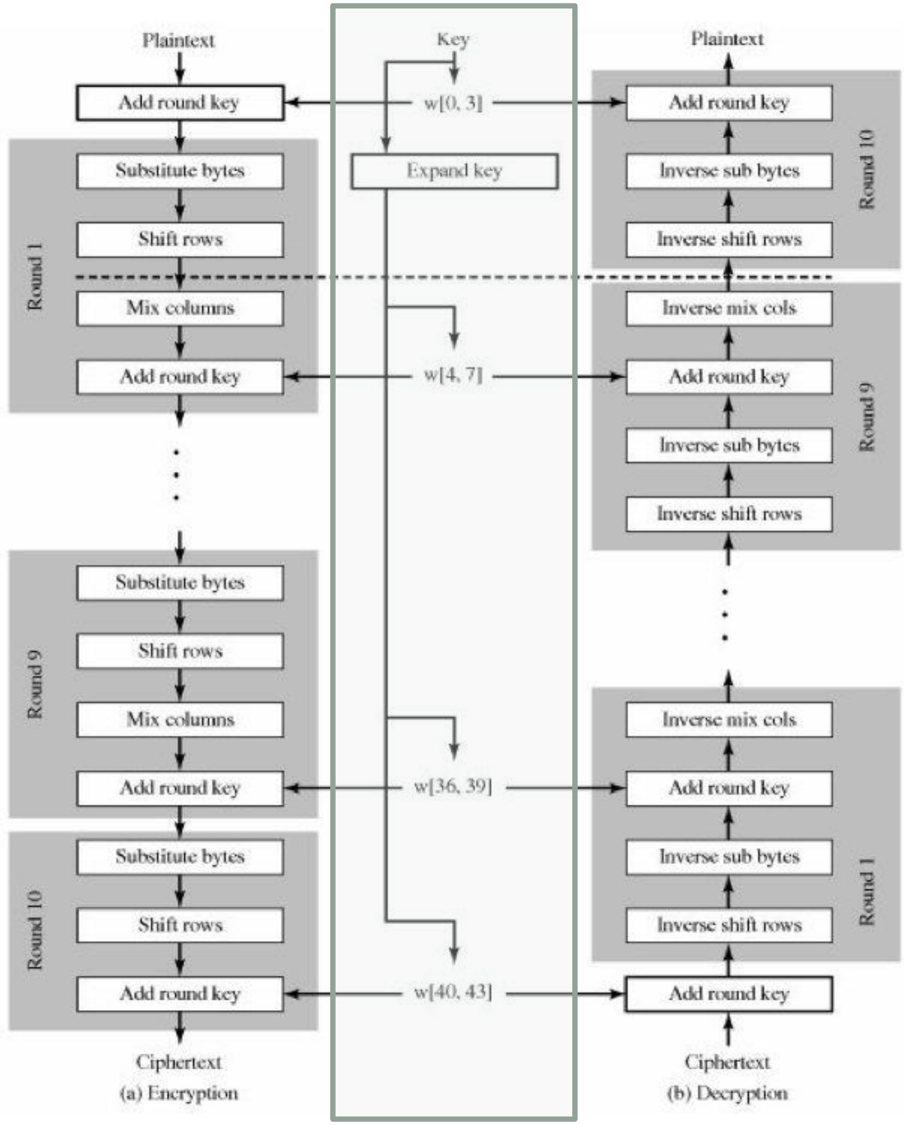
Key mixing (add round)

Addition of the first row of the previous example:

04:	00000100	40:	01000000
AC:	10101100 $\oplus$	19:	00011001 $\oplus$
A8:	10101000	59:	01011001
A3:	10100011	4C:	01001100
28:	00101000 $\oplus$	57:	01010111 $\oplus$
8B:	10001011	1B:	00011011


State
SubBytes
S
State
ShiftRows
State
MixColumns
M
State
AddRoundKey
r0 r1 r2 r3 r4 r5 r6 r7 r8 r9 r10 r11 r12 r13 r14 r15
State

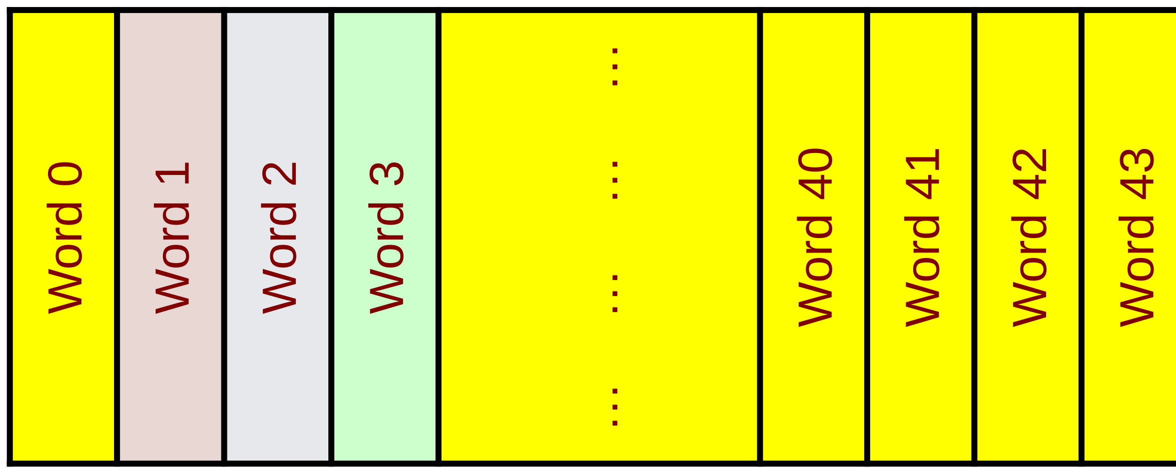
AES – Key expansion



Aes - Key Expansion

- AES key expansion algorithm takes as input a 4-word (16 bytes) key and produces a linear arrays of 44 words (176 bytes).

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}



Initial 4-word keys 10 x 4-word round keys

Aes - Key Expansion

1. The key is copied into the first four words of the expanded key.
2. The remainder ten 4-word round keys are generated using the following key expansion algorithm:
   i. Each word w[i] is set to the sum modulo 2 of the previous word w[i-1] and the word that is located N_k (key size) positions earlier; i.e., w[i-N_k].
   ii. For words in positions that are a multiple of N_k, a transformation (word rotation and byte substitution) is applied to w[i-1] prior to the addition modulo 2 to the word that is located N_k position earlier, followed by an addition modulo 2 with the round constant word.

Aes - Key Expansion

- The word rotation operation is to left-rotate the round-key $\{w_0, w_1, w_2, w_3\}$ one position to form the word $\{w_1, w_2, w_3, w_0\}$. We name this operation Rotword().
- The byte substitution is done on the rotated word $\{w_1, w_2, w_3, w_0\}$ (the result of step i) using the AES S-box. We name this operation Subbyte().

Aes - Key Expansion

Round constant is fixed as the array of powers of 2 in $GF(2_8)$

- The round constant in Hexadecimal:

Round constant 1	01	00	00	00
Round constant 2	02	00	00	00
Round constant 3	04	00	00	00
Round constant 4	08	00	00	00
Round constant 5	10	00	00	00
Round constant 6	20	00	00	00
Round constant 7	40	00	00	00
Round constant 8	80	00	00	00
Round constant 9	1B	00	00	00
Round constant 10	36	00	00	00

Aes - Key Expansion

For example:

AC	19	28	57
77	FA	D1	5C
66	DC	29	00
F3	21	41	6A

$W_{(i-Nk)}$			$W_{(i-1)}$	W_i							
AC	19	28	57								
77	FA	D1	5C						…		
66	DC	29	00								
F3	21	41	6A								
W_0	W_1	W_2	W_3	W_4	W_5	W_6	W_7	W_8	…	W_{42}	W_{43}

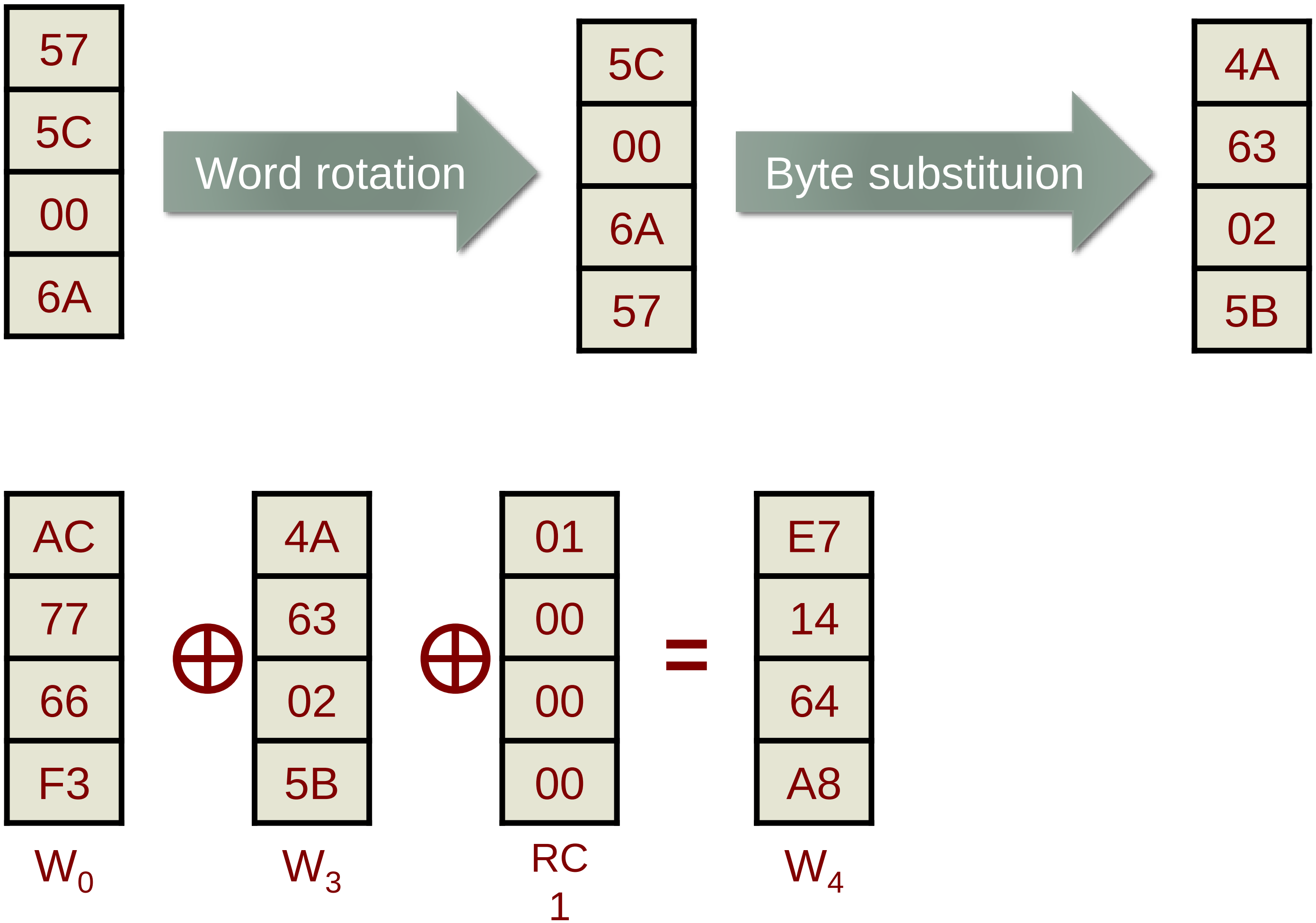
Aes - Key Expansion

$W_{(i-Nk)}$			$W_{(i-1)}$	W_i							
AC	19	28	57								
77	FA	D1	5C						…		
66	DC	29	00								
F3	21	41	6A								
W_0	W_1	W_2	W_3	W_4	W_5	W_6	W_7	W_8	…	W_{42}	W_{43}

- i = 4, and 4 is a multiple of N_k (N_k = 4) => $W_{i-1} = W_3$ is transformed with word rotation (Rotword()) and byte substitution (Subbyte()) before it is sum-modulo 2 with W_{i-Nk} and first round constant (RCon1).

$$W_4 = \text{Subbyte}(\ \text{Rotword}(W_3)\ ) \oplus W_0 \oplus \text{Round constant } 1$$

Aes - Key Expansion



Aes - Key Expansion

$W_{(i-Nk)}$			$W_{(i-1)}$	W_i							
AC	19	28	57	E7							
77	FA	D1	5C	14					…		
66	DC	29	00	64							
F3	21	41	6A	A8							
W_0	W_1	W_2	W_3	W_4	W_5	W_6	W_7	W_8	…	W_{42}	W_{43}

- i = 5, and 5 is not a multiple of N_k (N_k = 4)

$$W_5 = W_4 \oplus W_1$$

E7		19		FE
14	$\oplus$	FA	=	EE
64		DC		B8
A8		21		89
W_4		W_1		W_5

Aes - Key Expansion

$W_{(i-Nk)}$			$W_{(i-1)}$	W_i							
AC	19	28	57	E7	FE				…		
77	FA	D1	5C	14	EE						
66	DC	29	00	64	B8						
F3	21	41	6A	A8	89						
W_0	W_1	W_2	W_3	W_4	W_5	W_6	W_7	W_8	…	W_{42}	W_{43}

- i = 6, and 6 is not a multiple of N_k (N_k = 4)

$$W_6 = W_5 \oplus W_2$$

$$\underset{W_5}{\begin{bmatrix} \mathrm{FE} \\ \mathrm{EE} \\ \mathrm{B8} \\ 89 \end{bmatrix}} \oplus \underset{W_2}{\begin{bmatrix} 28 \\ \mathrm{D1} \\ 29 \\ 41 \end{bmatrix}} = \underset{W_6}{\begin{bmatrix} \mathrm{D6} \\ \mathrm{3F} \\ 91 \\ \mathrm{C8} \end{bmatrix}}$$

Aes - Key Expansion

$W_{(i-Nk)}$			$W_{(i-1)}$	W_i							
AC	19	28	57	E7	FE	D6					
77	FA	D1	5C	14	EE	3F			...		
66	DC	29	00	64	B8	91					
F3	21	41	6A	A8	89	C8					
W_0	W_1	W_2	W_3	W_4	W_5	W_6	W_7	W_8	...	W_{42}	W_{43}

- i = 7, and 7 is not a multiple of N_k ($N_k = 4$)

$$W_7 = W_6 \oplus W_3$$

W_6		W_3		W_7
D6		57		81
3F	$\oplus$	5C	=	63
91		00		91
C8		6A		A2

Aes - Key Expansion

$W_{(i-Nk)}$			$W_{(i-1)}$	W_i							
AC	19	28	57	E7	FE	D6	81				
77	FA	D1	5C	14	EE	3F	63		…		
66	DC	29	00	64	B8	91	91				
F3	21	41	6A	A8	89	C8	A2				
W_0	W_1	W_2	W_3	W_4	W_5	W_6	W_7	W_8	…	W_{42}	W_{43}

- As an exercise, complete the key for round 3; i.e., W_8 to W_{11}.

TUTORIAL 3

AES Related Questions

AES Related questions

Briefly describe the Shift Rows and Byte Substitution layers of AES. Explain why we can apply them in either order with the same result.

AES Related questions

The state in AES is a 4×4 matrix with entries in the field of 256 elements. The shift row layer shifts each row to the right a certain amount, wrapping the entries around. More precisely, the first row is not shifted, the second row is shifted by one, the third row is shifted by two, and the fourth row is shifted by three.

AES Related questions

The Byte Substitution layer can be viewed as a lookup table. Each matrix entry, represented by an 8-bit byte, is broken into two pieces which index the rows and columns of a 16 × 16 lookup matrix. The byte is replaced by the corresponding entry in the table, which is another 8-bit byte.

AES Related questions

The Byte Substitution layer is applied entry by entry to the state, with all entries treated in the same way. The Row Shift layer simply moves the bytes around. Thus it doesn't matter in which order we apply these layers: shifting and substituting is the same as first substituting then shifting.