Network Security and Cryptography
Symmetric-key cryptography

Lecture 6: AES

Mark Ryan

The Advanced Encryption Standard

AES is the "successor" of DES. Like DES, it is a block cipher; but it has a larger block size, and a larger key size.

Rationale for replacing DES: DES considered insecure; 3DES considered too slow.

Process for replacing DES: A NIST competition was held in 1997
15 submissions 1998; 5 finalists 1999
Rijndael was winner, named after its two inventors, two Belgian cryptographers, Vincent Rijmen and Joan Daemen.

Rijndael was adopted as the recommended successor to DES in 2000, and is now called AES.

AES parametrisable:

- Block size 128
- key sizes of 128, 192 and 256 bits
- 10, 12 or 14 rounds of encryption for each of those key sizes

Similarly to DES, AES works in rounds, with round keys.
Here, we look at AES-128.

AES is a substitution-permutation network (not a Feistel network).
Start by arranging the message in 4×4 matrix of 8-bit elements, filling it downwards and then right
Each round has following operations:

- Substitution: Operating on every single byte independently. This gives the *non-linearity* in AES.
- Byte permutation ShiftRows
- Column manipulation MixColumns. ShiftRows and MixColumns give us *diffusion* in AES.
- Xor with round key This provides the *key addition* in AES.

The 10 rounds are preceded by a key addition (thus, there are 11 key additions in total). The final round is slightly simpler: there's no MixColumns.

Byte operations in AES

AES is a byte-oriented cipher. The 128 bit "state" which is mapulated by the rounds is considered as 16 bytes, arranged in a matrix:

$$\begin{bmatrix} A_0 & A_4 & A_8 & A_{12} \\ A_1 & A_5 & A_9 & A_{13} \\ A_2 & A_6 & A_{10} & A_{14} \\ A_3 & A_7 & A_{11} & A_{15} \end{bmatrix}$$

To define the operations used in AES, we need two operations on bytes: $\oplus$ and $\otimes$. Each of those operations takes two bytes, and returns another byte. For example,

$$11000010 \oplus 00101111 = 11101101 \text{ and } 11000010 \otimes 00101111 = 00000001$$

The operation $\oplus$ is just bitwise-xor. The operation $\otimes$ on 8-bit numbers is called multiplication in $\mathbb{F}_{2^8}$. This is quite a difficult operation to program from scratch (we will define it later). In most implementations, $\otimes$ is done as a lookup table in code.

Substitution

Each byte in the current 4x4 state matrix is used as an index to the S-box, obtaining a new byte for that position.

The content of the S-box is mathematically defined (we will see that definition later). In most implementations, the S-box is implemented as a lookup table.

The S-box is shown on the following slide.

	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

Substitution

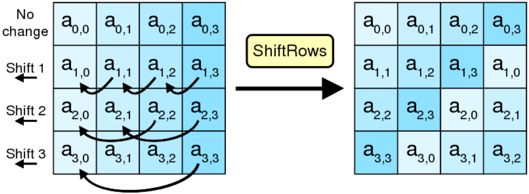
Unlike in the case of DES, the S-box isn't just an arbitrary look-up table.

We will see later that it is defined using a calculation in the field $\mathbb{F}_{2^8}$ (details later).

Implementation: done as a lookup table in code.

Shift Rows

ShiftRows performs cyclic shift on the state matrix



Source: Wikipedia

MixColumns

Mixing each column separately
Achieved by multiplying with matrix

$$\begin{bmatrix} b_{0,i} \\ b_{1,i} \\ b_{2,i} \\ b_{3,i} \end{bmatrix} = \begin{bmatrix} 0x02 & 0x03 & 0x01 & 0x01 \\ 0x01 & 0x02 & 0x03 & 0x01 \\ 0x01 & 0x01 & 0x02 & 0x03 \\ 0x03 & 0x01 & 0x01 & 0x02 \end{bmatrix} \cdot \begin{bmatrix} a_{0,i} \\ a_{1,i} \\ a_{2,i} \\ a_{3,i} \end{bmatrix}$$

In this matrix multiplication, we use $\oplus$ (xor) for addition, and the previously-mentioned "special" operation $\otimes$ for multiplication.

Adding Round Key

Key is 128 bits

Key schedule is used to compute 10x 128-bit round keys

The round keys can also be represented as 4×4 matrix.
Simply xor'ed to state matrix.

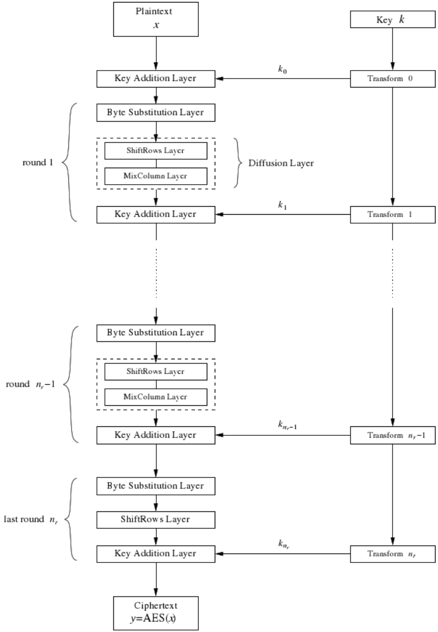
Key schedule

Derive round keys K_i as follows:
Split K into four words W_0, W_1, W_2 and W_3 of 32 bits each.

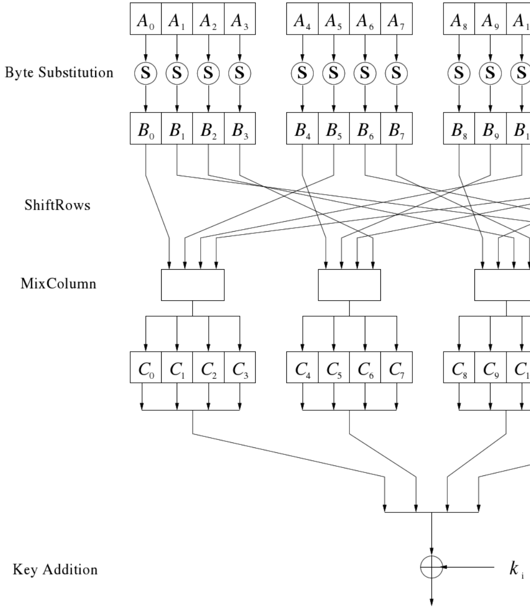
$$
\begin{array}{l}
\textbf{for } i := 1 \textbf{ to } 10 \textbf{ do} \\
\quad T := W_{4i-1} \lll 8 \\
\quad T := \text{SubBytes}(T) \\
\quad T := T \oplus RC_i \\
\quad W_{4i} := W_{4i-4} \oplus T \\
\quad W_{4i+1} := W_{4i-3} \oplus W_{4i} \\
\quad W_{4i+2} := W_{4i-2} \oplus W_{4i+1} \\
\quad W_{4i+3} := W_{4i-1} \oplus W_{4i+2} \\
\textbf{end}
\end{array}
$$

Here, RC_i are 32-bit constants defined in AES (we will see their exact definition later).
The round keys K_i are obtained as follows:

$$K_i = W_{4i}, W_{4i+1}, W_{4i+2}, W_{4i+3}.$$



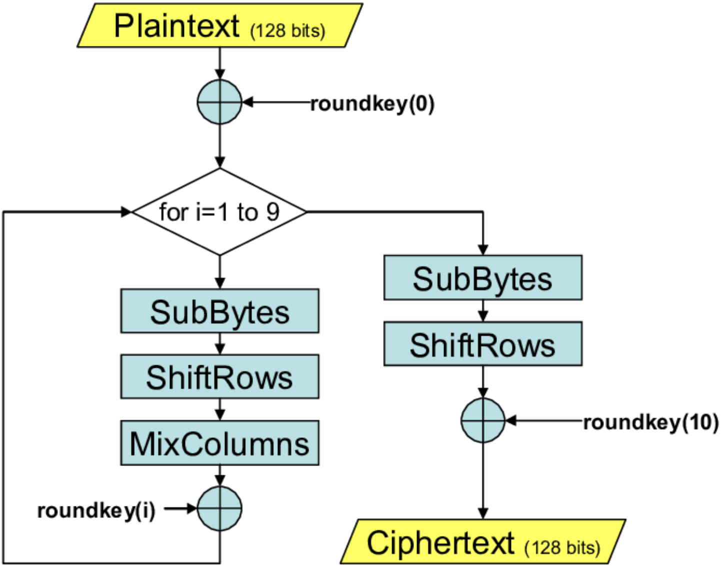
Plaintext
x
Key k
Key Addition Layer
k_0
Transform 0
round 1
Byte Substitution Layer
ShiftRows Layer
MixColumn Layer
Diffusion Layer
Key Addition Layer
k_1
Transform 1
round $n_r - 1$
Byte Substitution Layer
ShiftRows Layer
MixColumn Layer
Key Addition Layer
$k_{n_r - 1}$
Transform $n_r - 1$
last round n_r
Byte Substitution Layer
ShiftRows Layer
Key Addition Layer
k_{n_r}
Transform n_r
Ciphertext
y=AES(x)



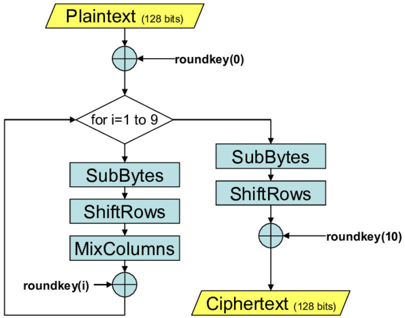
A_0 A_1 A_2 A_3 A_4 A_5 A_6 A_7 A_8 A_9 A_{10} A_{11} A_{12} A_{13} A_{14} A_{15}
Byte Substitution
S
B_0 B_1 B_2 B_3 B_4 B_5 B_6 B_7 B_8 B_9 B_{10} B_{11} B_{12} B_{13} B_{14} B_{15}
ShiftRows
MixColumn
C_0 C_1 C_2 C_3 C_4 C_5 C_6 C_7 C_8 C_9 C_{10} C_{11} C_{12} C_{13} C_{14} C_{15}
Key Addition
k_i

AES on a single slide?

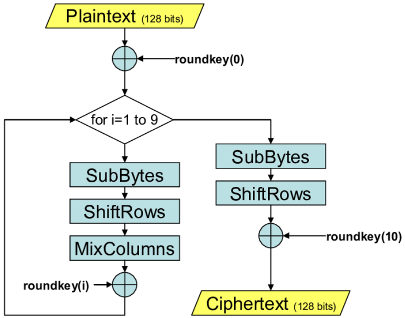
AES (Advanced Encryption Standard, 2001)

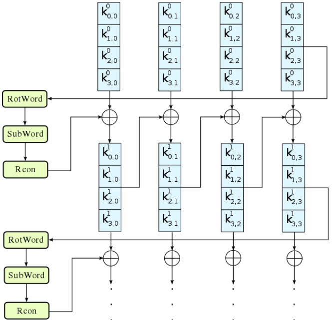


AES (Advanced Encryption Standard, 2001)

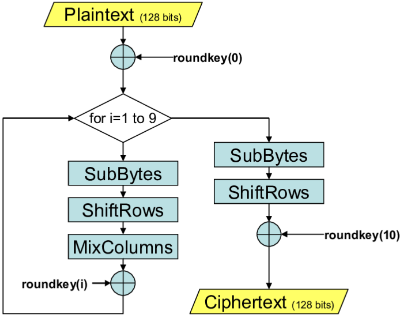


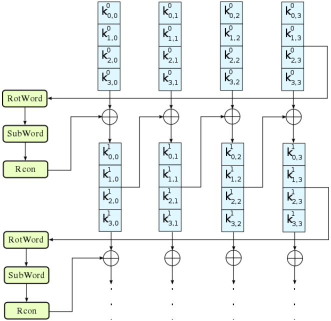
AES (Advanced Encryption Standard, 2001)

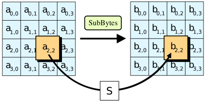




AES (Advanced Encryption Standard, 2001)

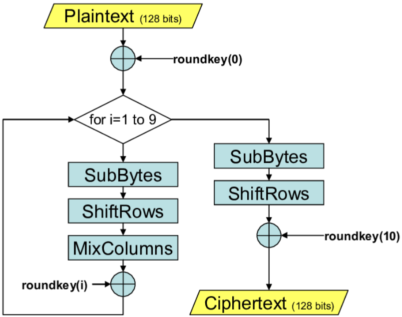


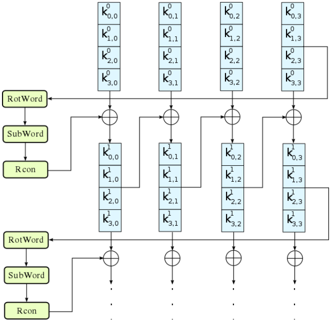


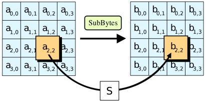


Sub-bytes

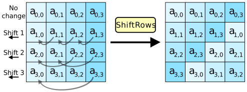
AES (Advanced Encryption Standard, 2001)





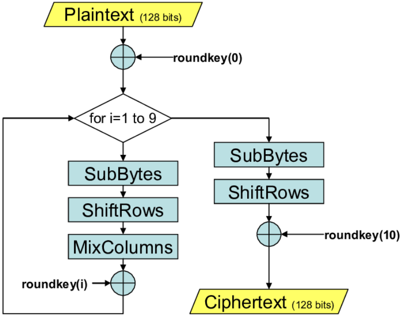


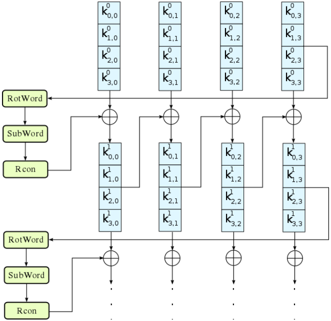
Sub-bytes

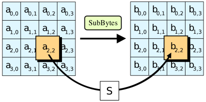


Shift-rows

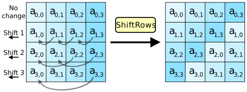
AES (Advanced Encryption Standard, 2001)



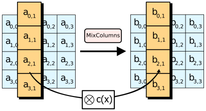




Sub-bytes

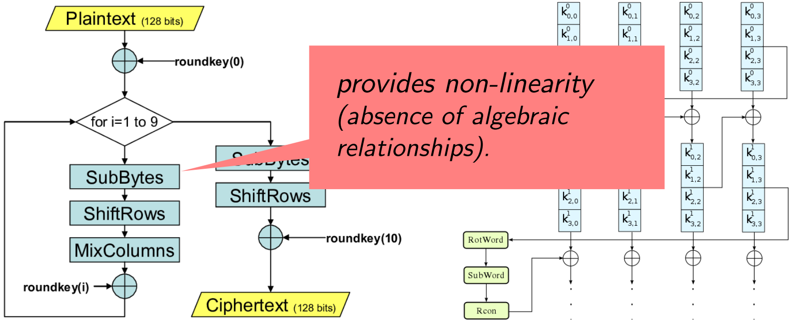


Shift-rows

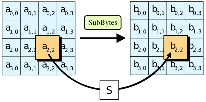


Mix-cols

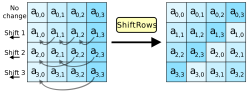
AES (Advanced Encryption Standard, 2001)



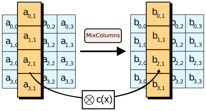
provides non-linearity (absence of algebraic relationships).



Sub-bytes

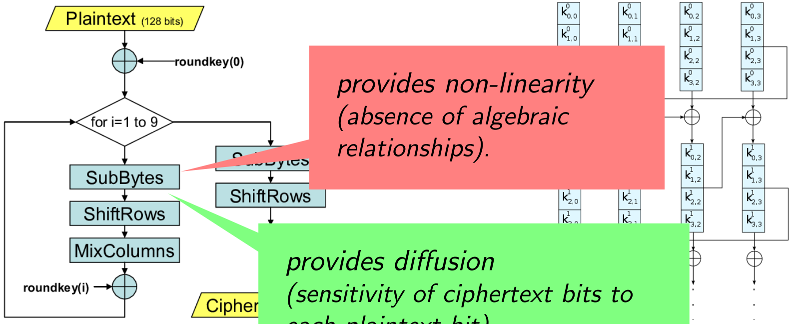


Shift-rows



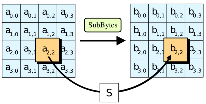
Mix-cols

AES (Advanced Encryption Standard, 2001)

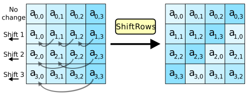


provides non-linearity (absence of algebraic relationships).

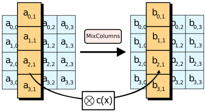
provides diffusion (sensitivity of ciphertext bits to each plaintext bit).



Sub-bytes

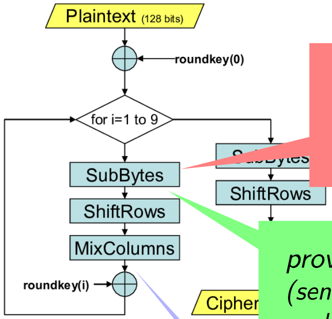


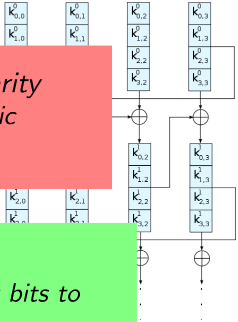
Shift-rows



Mix-cols

AES (Advanced Encryption Standard, 2001)

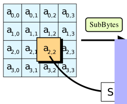




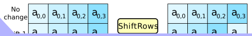
provides non-linearity (absence of algebraic relationships).

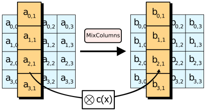
provides diffusion (sensitivity of ciphertext bits to each plaintext bit).

provides confusion (sensitivity of ciphertext bits to each key bit).



Sub-b





Mix-cols

AES security

Still considered to have very good security. The main known attack is a "related key" attack: if the attacker knows a key, and knows that you are using a "related" key, then some information leakage may occur. If AES is used correctly, keys are always chosen randomly, and therefore are never "related". So in that case, this has no practical significance.