Introduction to Information Security

Christian J. Rudder

November 2024

Contents

This page is left intentionally blank.

Big thanks to **Professor Sharon Goldberg**
for teaching CS357 (Introduction to Information Security)
at Boston University.

The following five sections are pre-requisites from Concise Work Section Modules [4].
Available at: https://github.com/Concise-Works/sect-modules.
These are not needed, but are recommended for a better understanding of the material.

Now to discuss a possible shared key (symmetric key) algorithm that could be used as Enc_λ in GCM.

Definition 0.1: Advanced Encryption Standard (AES)

During a worldwide competition in 2001, the National Institute of Standards and Technology (NIST) selected the Rijndael algorithm as the Advanced Encryption Standard (AES). AES is a symmetric key block cipher that encrypts plaintext (PT) in 128-bit blocks. AES works in rounds permuting the PT with partial keys generated from the initial key.

- **Key Expansion**: An initial key is expanded into a key schedule. These keys will be assigned to each round. The rounds needed depend on the initial key size:
  1. 128-bit key: 10 rounds, 11 keys.
  2. 192-bit key: 12 rounds, 13 keys.
  3. 256-bit key: 14 rounds, 15 keys.
- **Input Transformation**: The PT is transformed into a 4×4 matrix, called the **State**. The state undergoes four main operations per round:
  1. **SubBytes**: Each byte is substituted with a value from the S-Box.
  2. **ShiftRows**: Each row is shifted left by an offset.
  3. **MixColumns**: Each column is mixed with a fixed matrix.
  4. **AddRoundKey**: Each byte in the State is XORed with a sub-key. [2] [3] [1]

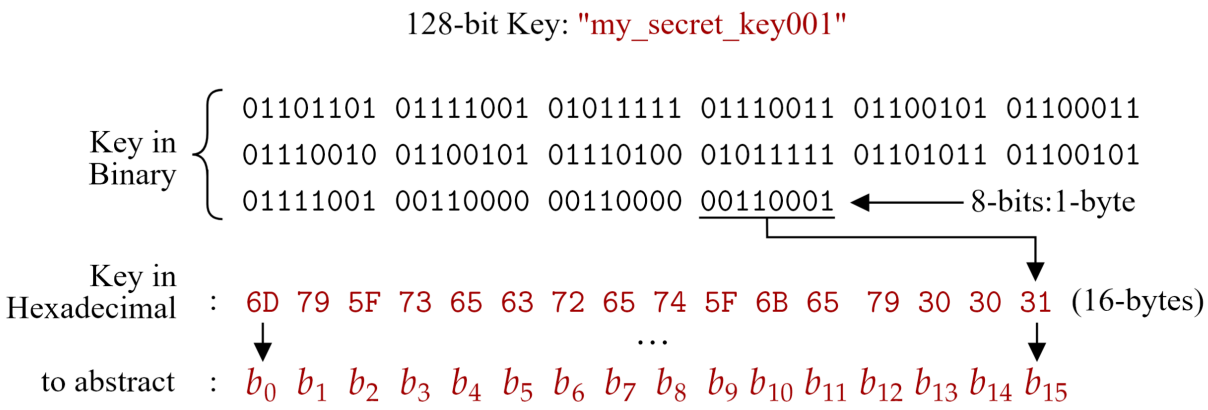
Definition 0.2: Key Expansion

Key expansion, an AES process which takes a single key and expands it into multiple keys. The initial key is broken up into 16-byte 4×4 matrices. Each column a **Word** (32-bits). The process follows four main steps:

RotWord→SubWord→Rcon→XOR.
(Rotate Word, Substitute Word, Round Constant, XOR)

This generates a **Sub-Key** for one round. Each round generates for the next round.

AES Key Expansion: Given the an initial 128-bit key, λ_0 "my_secret_key001":



Key represented an a 4×4 matrix

$$\begin{bmatrix} b_0 & b_4 & b_8 & b_{12} \\ b_1 & b_5 & b_9 & b_{13} \\ b_2 & b_6 & b_{10} & b_{13} \\ b_3 & b_7 & b_{11} & b_{15} \end{bmatrix}$$

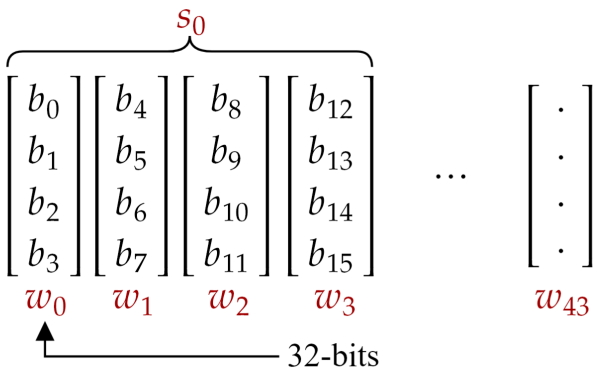
The Key is broken up into Words

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} \begin{bmatrix} b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} \begin{bmatrix} b_8 \\ b_9 \\ b_{10} \\ b_{11} \end{bmatrix} \begin{bmatrix} b_{12} \\ b_{13} \\ b_{14} \\ b_{15} \end{bmatrix}$$

w_0 w_1 w_2 w_3

These Words w_0, w_1, w_2, w_3 are the initial key. This will then generate the rest of the Words needed. This first group is the first sub-key s_0 for the first round.

This is a Subkey (s_0), used for a round. The number of rounds depend on the initial Key.



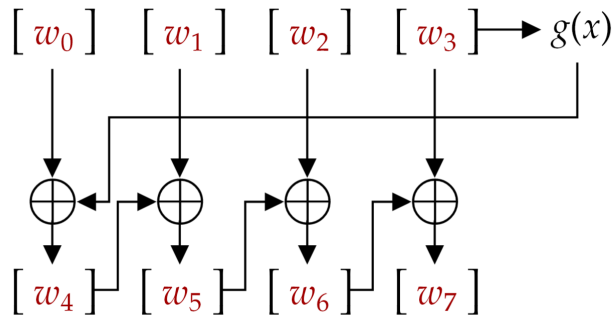
Round Dependence:

- 128-bit key: 10 rounds, 11 keys.
- 192-bit key: 12 rounds, 13 keys.
- 256-bit key: 14 rounds, 15 keys.

Hence, $4 \times 11 = 44$ Words

Rounds depend on $\|\lambda_0\|$: 128-bits $\rightarrow$ 10 rounds, 192-bits $\rightarrow$ 12 rounds, 256-bits $\rightarrow$ 14 rounds.

There need be a total of 44 Words. Expansion is used to find the rest:



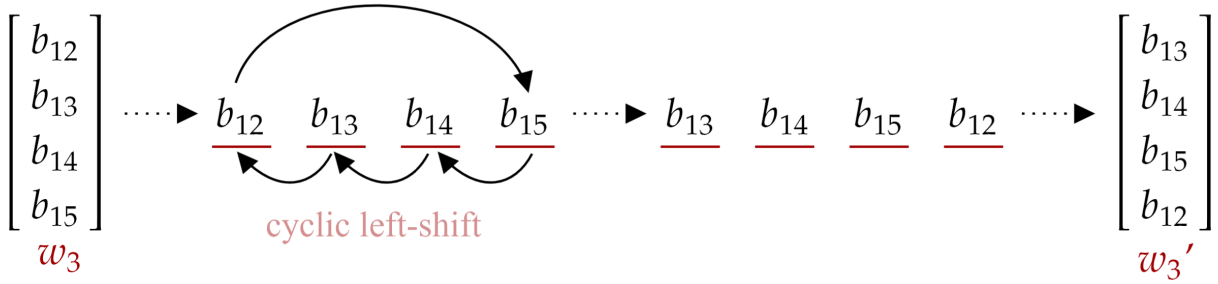
Here s_0 generates s_1

And so forth s_i generates s_{i+1}
Until all Words are satisfied

And $g(x)$ some permuting function

The $g(x) := RotWord \circ SubWord \circ Rcon \circ XOR$, generating the next sub-key s_1.

Here, $g(x)$ performs a cyclic left-shift on w_3; Often called $RotWord(x)$ (rotate Word).



Definition 0.3: Nibble

A **nibble** is a four-bit aggregation, referring to "half a **byte**." A nibble splits 8-bits into two 4-bit values, the **upper nibble** and the **lower nibble**. E.g., given the hex value $0A$, the upper and lower nibble is 0 and A respectively.

Now to quickly introduce some concepts needed for the AES borrowed from group theory.

Definition 0.4: Field

A **field** is a set of elements equipped with two operations, addition and multiplication, such that the following properties hold:

- **Closure**: Adding or multiplying any two elements in the set remains in the set.
- **Associativity**: Addition and multiplication are associative.
- **Commutativity**: Addition and multiplication are commutative.
- **Identities**: There exist identity elements for addition (0) and multiplication (1).
- **Inverses**: Every element has an additive inverse (negative) and, except for 0, a multiplicative inverse (reciprocal).
- **Distributivity**: Multiplication distributes over addition.

E.g., The set of real numbers $\mathbb{R}$, with standard addition and multiplication, forms a field.

Theorem 0.1: Abel Ruffini Theorem

There is no general for solving polynomials of degree five or higher. Moreover, no formula is possible using these finite amount of: $+, -, \times, \div, \sqrt[n]{x}$.

This will play a key part in the AES algorithm, stopping attackers from reversing the encryption process.

Definition 0.5: Galois Field $GF(2^8)$

The Galois Field $GF(2^8)$ is a finite field consisting of $2^8 = 256$ elements, where each element is an 8-bit binary value (a byte). Addition and multiplication in $GF(2^8)$ are defined modulo an irreducible polynomial of degree 8 over $GF(2)$ (the binary field).
In AES, the irreducible polynomial used is:

$$p(x) = x^8 + x^4 + x^3 + x + 1.$$

Tip: To learn more consider this video on Galois Theory: "Why is there no quintic formula?"

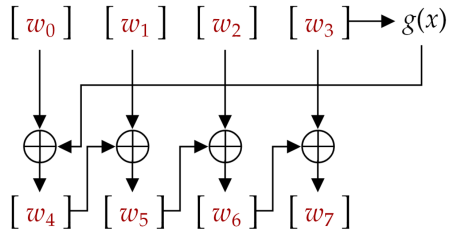
Definition 0.6: AES S-Box

The AES S-Box is a substitution box used in the AES algorithm. It is a 16×16 matrix of 8-bit values. The first column and row represent the upper and lower nibble of the input byte. E.g., given $C7$, column $c0$ and row 07 intersect $c6$.

	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	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08	ba
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	89	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

For reference, previous figures are shown again.

There need be a total of 44 Words. Expansion is used to find the rest:

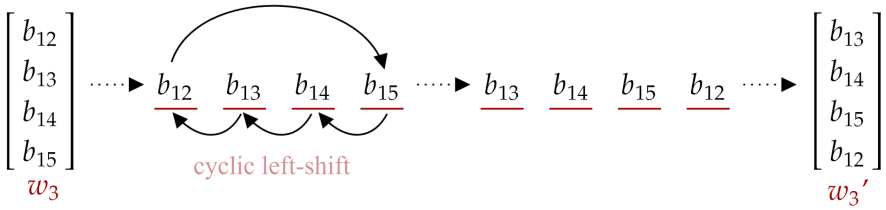


Here s_0 generates s_1

And so forth s_i generates s_{i+1}
Until all Words are satisfied

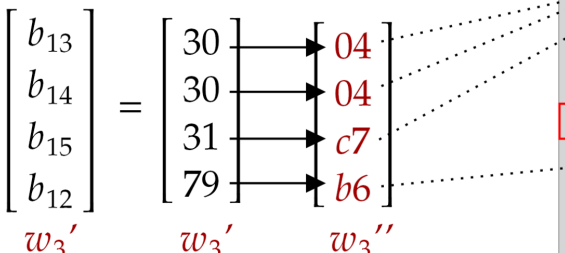
And $g(x)$ some permuting function

Here, $g(x)$ performs a cyclic left-shift on w_3; Often called $RotWord(x)$ (rotate Word).



Next $SubWord(x)$ takes the Hex digit places of b_i to index the **AES S-Box**

Key Hex : 6D 79 5F 73 65 63 72 65 74 5F 6B 65 79 30 30 31



	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	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08	ba
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	89	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

Definition 0.7: Round Constants

Round constants take the output of $SubWord(x)$ and XOR the first byte with a constant value. The constant value, denoted $Rcon[j]$, is derived from powers of 2 in the Galois Field $GF(2^8)$, where the irreducible polynomial is $x^8 + x^4 + x^3 + x + 1$.

Each round constant ensures the uniqueness of round keys during the AES key expansion process, introducing nonlinearity and breaking symmetry between rounds.

Round (i)	***Rcon* Value**	**Power of 2 in GF(2^8)**
1	0x01 00 00 00	$2^0 = 1$
2	0x02 00 00 00	$2^1 = 2$
3	0x04 00 00 00	$2^2 = 4$
4	0x08 00 00 00	$2^3 = 8$
5	0x10 00 00 00	$2^4 = 16$
6	0x20 00 00 00	$2^5 = 32$
7	0x40 00 00 00	$2^6 = 64$
8	0x80 00 00 00	$2^7 = 128$
9	0x1B 00 00 00	2^8 mod poly
10	0x36 00 00 00	2^9 mod poly

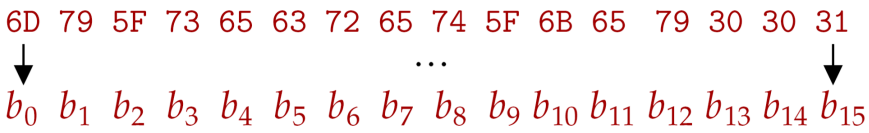
Table 1.1: Round Constants, where poly:= $x^8 + x^4 + x^3 + x + 1$.

Finally w_3'' is XORed with it's respective round constant $Rcon[j]$ from $GF(2^8)$.

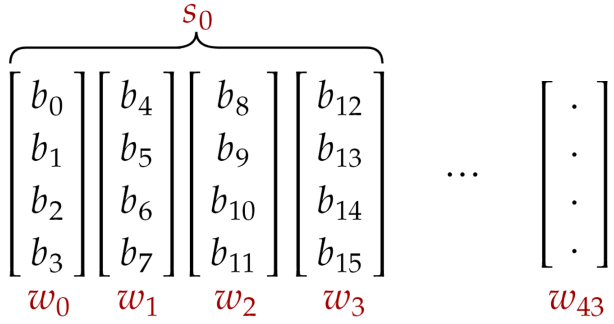
$$
\begin{aligned}
w_3'' &= 0000\ 0100\ 0000\ 0100\ 1100\ 0111\ 1011\ 0110 \\
Rcon[1] &= 0000\ 0001\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000 \\
\hline
w_3'' \oplus Rcon[1] &= 0000\ 0101\ 0000\ 0100\ 1100\ 0111\ 1011\ 0110
\end{aligned}
\quad \rightarrow \texttt{05 04 C7 B6} = \begin{bmatrix} 05 \\ 04 \\ C7 \\ B6 \end{bmatrix} g(w_3)
$$

All together:

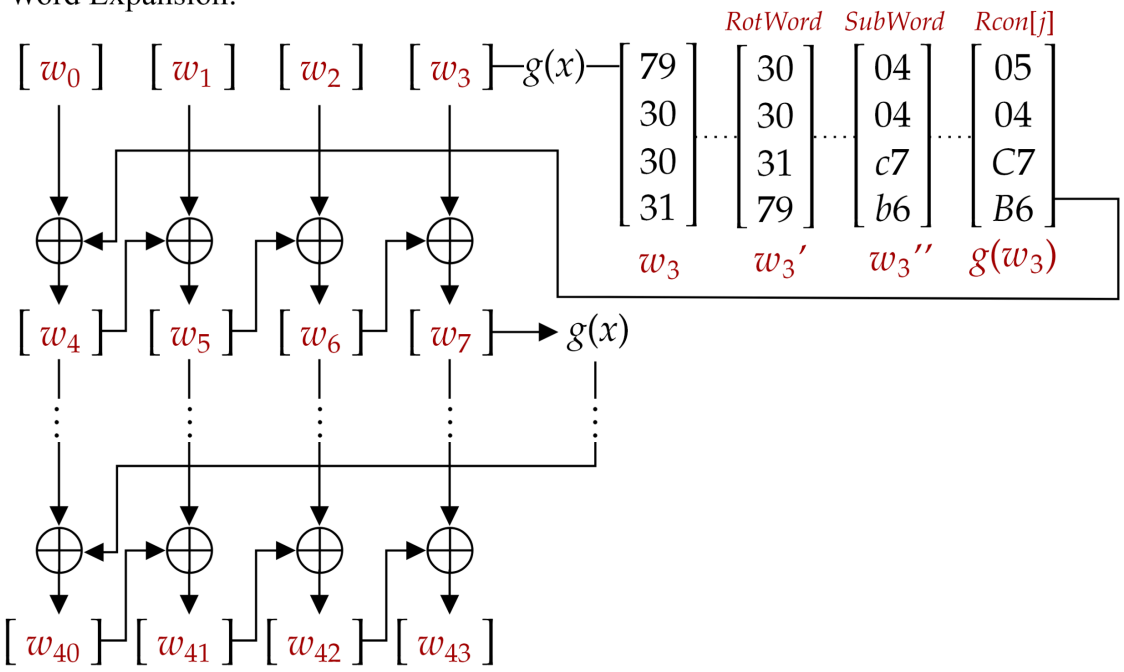
AES Key Expansion of a 128-bit Key: "my_secret_key001". First, Hex conversion:



Word Aggregation into Subkeys:



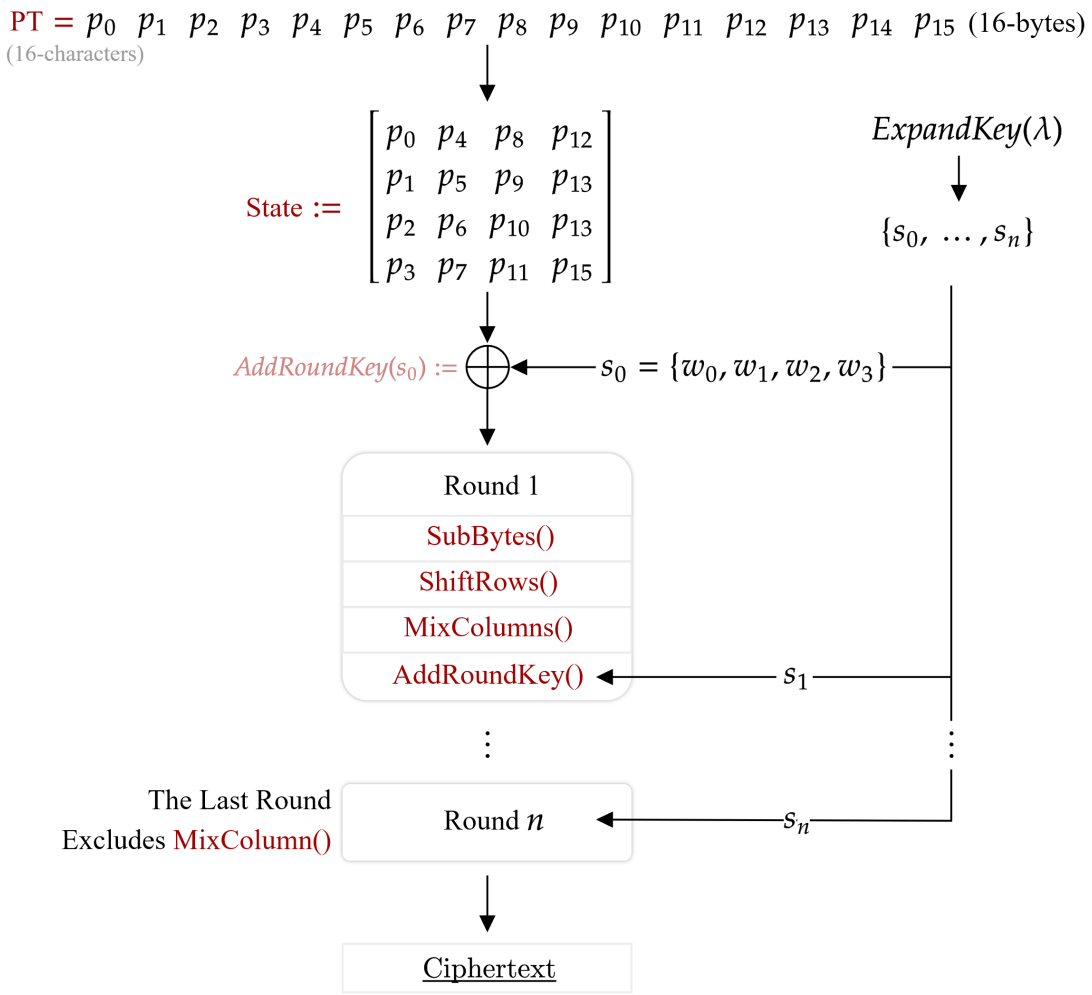
Word Expansion:



This is AES Key Expansion.

Now to discuss the AES encryption process.

The Four main steps in AES Encryption of a 128-bit Plaintext (PT) with a 128-bit key (λ).

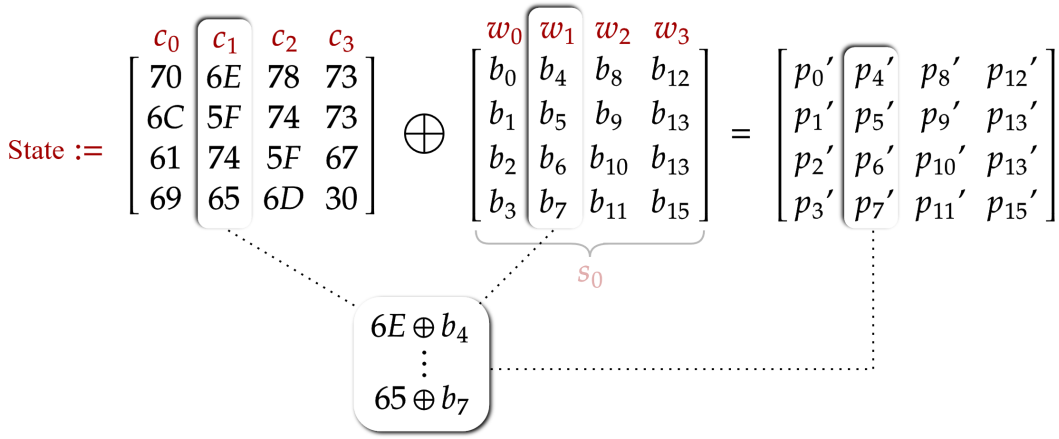


First the PT is transformed into a 4×4 matrix, called the **State**. The initial transformation is the **AddRoundKey** operation, which XORs each column of the state with sub-key $s_0 = \{w_0, w_1, w_2, w_3\}$. I.e., $[p_0, p_1, p_2, p_3] \oplus [w_0]$ and so on. Recall $w_0 = [b_0, b_1, b_2, b_3]$ for some byte b_i of the initial key.

To elaborate on the AddRoundKey operation:

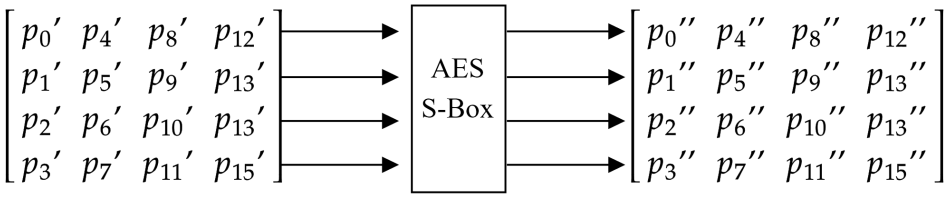
The AddRoundKey() XORs each column c_i with a sub-key Word w_i

PT = "plain_text_mssg0" Hex = `70 6C 61 69 6E 5F 74 65 78 74 5F 6D 73 73 67 30`



AddRoundKey takes a sub-key s_i and XORs each Word w_i with the corresponding column c_i of the State.

The SubBytes() performs an **AES S-Box** substitution with every byte of the State.



Just like in Key Expansion's (0.2) use of the S-Box on the last word of each sub-key. The PT undergoes a full substitution with the S-Box.

The ShiftRows() a fixed number of left-shifts depending on the row r_i of the State.

$S(r_i, x)$:= shifts row r_i to the left by x places (for ease of notation)

$$\begin{bmatrix} p_0'' & p_4'' & p_8'' & p_{12}'' \\ p_1'' & p_5'' & p_9'' & p_{13}'' \\ p_2'' & p_6'' & p_{10}'' & p_{13}'' \\ p_3'' & p_7'' & p_{11}'' & p_{15}'' \end{bmatrix} \begin{matrix} \longrightarrow S(r_0, 0) \longrightarrow \\ \longrightarrow S(r_1, 1) \longrightarrow \\ \longrightarrow S(r_2, 2) \longrightarrow \\ \longrightarrow S(r_3, 3) \longrightarrow \end{matrix} \begin{bmatrix} p_0'' & p_4'' & p_8'' & p_{12}'' \\ p_5'' & p_9'' & p_{13}'' & p_1'' \\ p_{10}'' & p_{13}'' & p_2'' & p_6'' \\ p_{15}'' & p_3'' & p_7'' & p_{11}'' \end{bmatrix}$$

Below, the notation {01} refers to a hex value.

In MixColumns(), Columns c_i are considered four-term polynomials of $GF(2^8)$ and multiplied by polynomial $a(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\}$

$$\vdots$$

$$= \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \quad \text{(polynomial matrix representation)}$$

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \times \begin{bmatrix} p_0'' \\ p_5'' \\ p_{10}'' \\ p_{15}'' \end{bmatrix} = \begin{bmatrix} p_0''' \\ p_5''' \\ p_{10}''' \\ p_{15}''' \end{bmatrix}$$

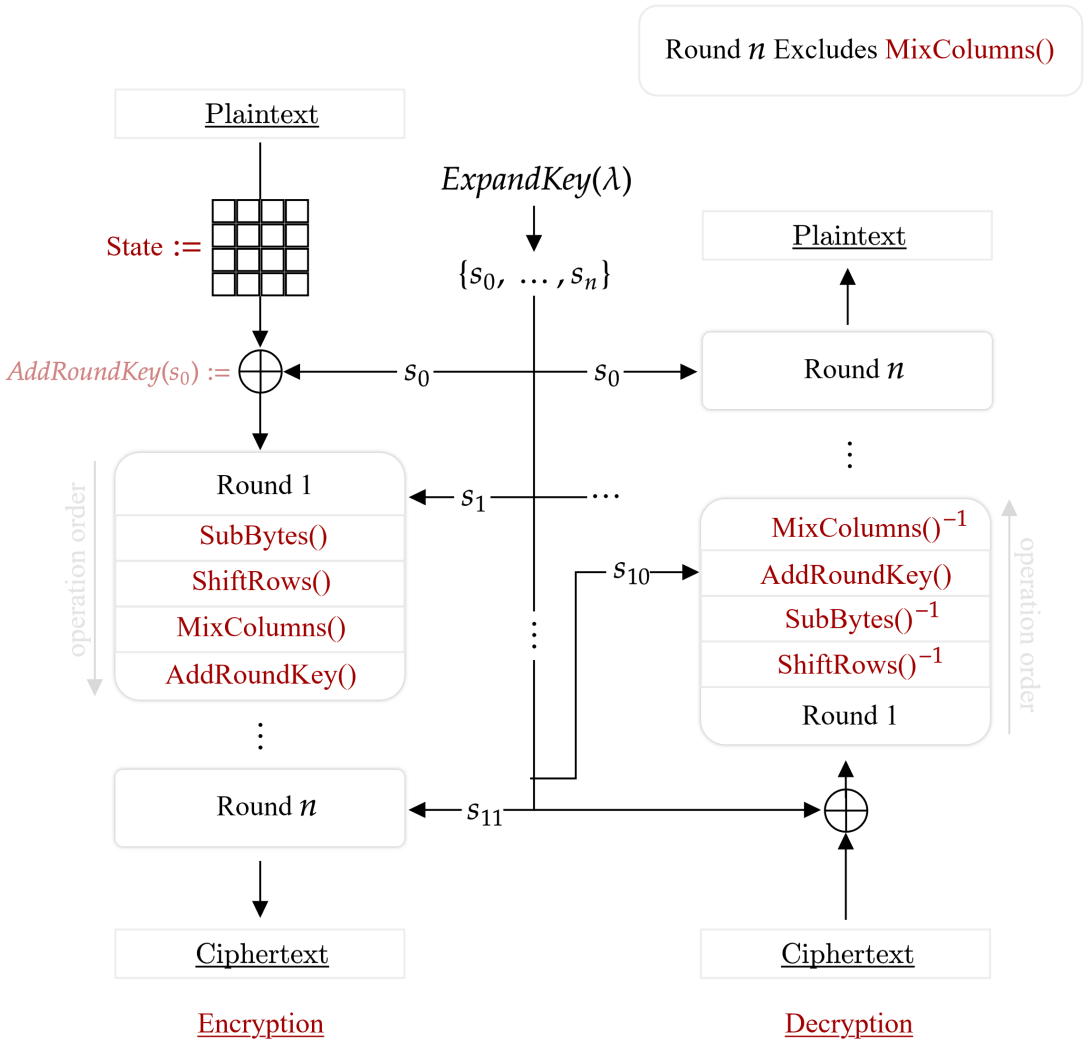
$$\vdots$$

$$p_0''' = (\{02\} \cdot p_0'') \oplus (\{03\} \cdot p_5'') \oplus p_{10}'' \oplus p_{15}''$$

$G(2^8)$, $a + b := a \oplus b$ (XOR)

$$\begin{bmatrix} p_0'' & p_4'' & p_8'' & p_{12}'' \\ p_5'' & p_9'' & p_{13}'' & p_1'' \\ p_{10}'' & p_{13}'' & p_2'' & p_6'' \\ p_{15}'' & p_3'' & p_7'' & p_{11}'' \end{bmatrix} \longrightarrow \text{MixColumns()} \longrightarrow \begin{bmatrix} p_0''' & p_4''' & p_8''' & p_{12}''' \\ p_5''' & p_9''' & p_{13}''' & p_1''' \\ p_{10}''' & p_{13}''' & p_2''' & p_6''' \\ p_{15}''' & p_3''' & p_7''' & p_{11}''' \end{bmatrix}$$

This is the AES Encryption & Decryption.



For Decryption, the last scheduled key for Encryption is used as the initial transformation. Note the inverse operations of Decryption follow a different order than Encryption, and use their respective inverse functions. This covers the basics of AES.

Bibliography

[1] Aes (advanced encryption standard) structure, 2018–2023. Copyright © 2018–2023 BrainKart.com; Developed by Therithal info, Chennai. Accessed: 2024-06-15.

[2] Satish CJ. Aes iii - advanced encryption standard - introduction, key expansion in aes cyber security cse4003, 2024. Accessed: 2024-06-15.

[3] National Institute of Standards and Technology (NIST), Morris J. Dworkin, Elaine Barker, James R. Nechvatal, James Foti, Lawrence E. Bassham, E. Roback, and James F. Dray Jr. Advanced encryption standard (aes). Federal Information Processing Standards Publication 197, National Institute of Standards and Technology, November 2001. Accessed: 2024-06-15. Local download available at https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=901427.

[4] Concise Works. Sect modules. https://github.com/Concise-Works/sect-modules, n.d. Accessed November 2024.