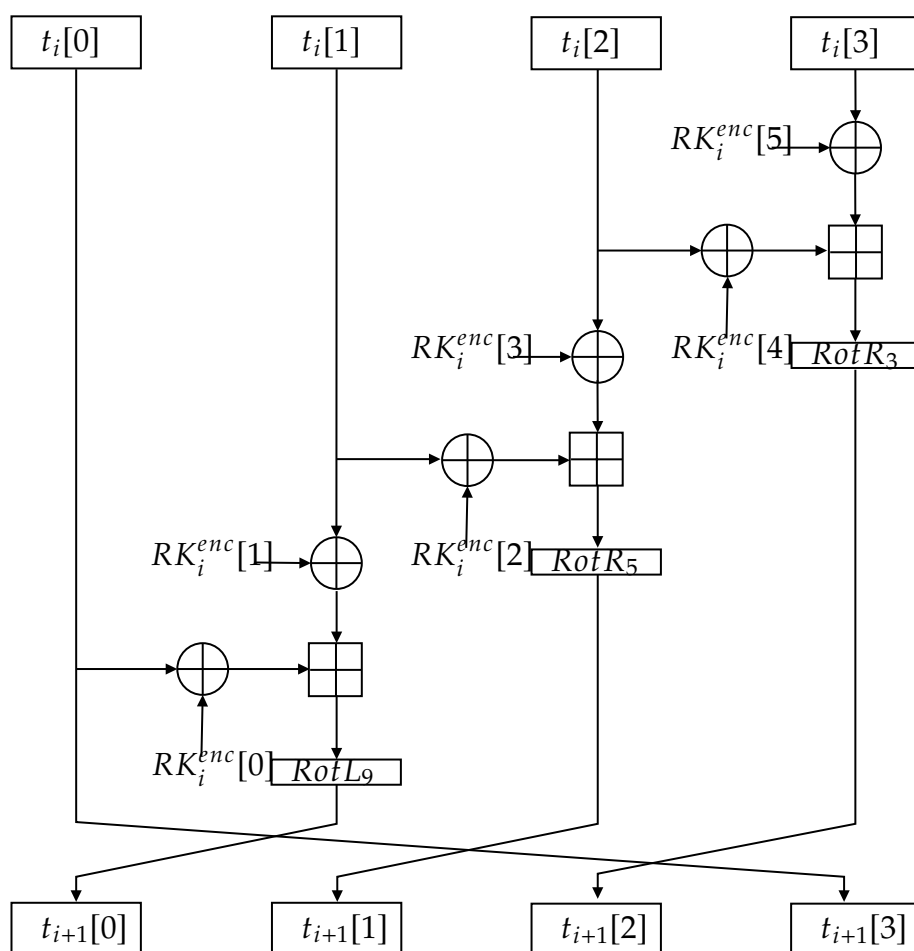


Lightweight Encryption Algorithm

- LEA -

Ji Yong-Hyeon



Department of Information Security, Cryptology, and Mathematics
 College of Science and Technology
 Kookmin University

January 10, 2024

List of Symbols

Contents

- 1 LEA: Implementation in Little-Endian 1**
 - 1.1 Specification 1
 - 1.2 State Representation 1
 - 1.3 Key Schedule 2
 - 1.3.1 Round Constant 3
 - 1.3.2 Rotation Function 3
 - 1.3.3 Encryption Key Schedule of LEA-128 3
 - 1.3.4 Decryption Key Schedule of LEA-128 4
 - 1.4 Encryption of LEA-128 5
 - 1.5 Decryption of LEA-128 6
- A Additional Data A 7**
 - A.1 Substitution-BOX 7

Chapter 1

LEA: Implementation in Little-Endian

1.1 Specification

Table 1.1: Specification Comparison between AES and LEA Block Ciphers

Specification	AES	LEA
Block Size (bits)	128	128
Key Size (bits)	128/192/256	128/192/256
Structure	Substitution-Permutation Network	Generalized Feistel Network
Rounds	10/12/14 (depends on key size)	24/28/32 (depends on key size)
Design Year	1998	2013

Table 1.2: Parameters of the Block Cipher LEA (1-word = 32-bit)

Algorithms	Block Size (N_b -byte)	Key Length (N_k -byte)	Number of Rounds (N_r)	Round-Key Length (byte)	Total Size of Round-Keys ($(N_r * 192)$ -bit)
LEA-128	16(4-word)	16(4-word)	24	24	4608 (144-word)
LEA-192	16(4-word)	24(6-word)	28	24	5376 (168-word)
LEA-256	16(4-word)	32(8-word)	32	24	6144 (192-word)

1.2 State Representation

Let $\text{state}[0], \text{state}[1], \dots$ be representation of arrays of bytes. Note that

$$\text{state}[i] := \{input_{8i}, input_{8i+1}, \dots, input_{8i+7}\} \in \mathbb{F}_{2^8}$$

for $input_i \in \mathbb{F}_2$. For example, $\text{state}[0] = \{input_0, input_1, \dots, input_7\}$.

The 128-bit plaintext P of LEA is represented as an array of four 32-bit words $P[0], P[1], P[2]$ and $P[3]$. Then

$$P[i] = \text{state}[4i + 3] \parallel \text{state}[4i + 2] \parallel \text{state}[4i + 1] \parallel \text{state}[4i] \quad \text{for } 0 \leq i \leq 3.$$

Here, $P[i] \in \mathbb{F}_{2^{32=8 \cdot 4}}$. The key K of LEA is also represented as the same way.

Table 1.3: Representations for words, bytes, and bits

Input Bit Sequence	24	...	31	16	...	23	8	...	15	0	...	7
Word Number	0											
Byte Number	3			2			1			0		
Bit Numbers in Word	31	...										1

Example 1.1.

128-bit Input String	0x0f1e2d3c4b5a69788796a5b4c3d2e1f0											
Split into Words	0x0f1e2d3c			0x4b5a6978			0x8796a5b4			0xc3d2e1f0		
	P[0]			P[1]			P[2]			P[3]		
P[0] (Word)	0x0f1e2d3c											
P[0] (Bit)	0b 0000:1111:0001:1110:0010:1101:0011:1010											
Split into Bytes	0x0f			0x1e			0x2d			0x3c		
	state[3]			state[2]			state[1]			state[0]		
state[0] (Byte)	0x3c											
Split into Bits	1111:0000			-			-			-		
	24	...	31	16	...	23	8	...	15	0	...	7

```

1  const char* inputString = "0f1e2d3c4b5a69788796a5b4c3d2e1f0";
2  u32 key[4];
3  stringToWordArray(inputString, key);
4
5  /*
6  (gdb) x/32xb key
7  0x7fffffffdd9c0: 0x3c    0x2d    0x1e    0x0f
8                    0x78    0x69    0x5a    0x4b
9  0x7fffffffdd9c8: 0xb4    0xa5    0x96    0x87
10                   0xf0    0xe1    0xd2    0xc3
11  0x7fffffffdd9d0: 0x01    0x00    0x00    0x00
12                   0x00    0x00    0x00    0x00
13  0x7fffffffdd9d8: 0xf6    0x75    0xae    0x03
14                   0x01    0x00    0x00    0x00
15  */

```

1.3 Key Schedule

$$\text{KeySchedule}_{128}^{\text{enc}} : \{0, 1\}^{128=8 \cdot 16} \rightarrow \{0, 1\}^{4608=192 \cdot 24}$$

$$\text{KeySchedule}_{192}^{\text{enc}} : \{0, 1\}^{192=8 \cdot 24} \rightarrow \{0, 1\}^{5376=192 \cdot 28}$$

$$\text{KeySchedule}_{256}^{\text{enc}} : \{0, 1\}^{256=8 \cdot 32} \rightarrow \{0, 1\}^{6144=192 \cdot 24}$$

1.3.1 Round Constant

The constant $\delta[i] \in \mathbb{F}_{2^{32}}$ ($i \in \{1, \dots, 7\}$) is as follows:

i	$\delta[i]$	value
0	$\delta[0]$	0xc3efe9db
1	$\delta[1]$	0x44626b02
2	$\delta[2]$	0x79e27c8a
3	$\delta[3]$	0x78df30ec
4	$\delta[4]$	0x715ea49e
5	$\delta[5]$	0xc785da0a
6	$\delta[6]$	0xe04ef22a
7	$\delta[7]$	0xe5c40957

1.3.2 Rotation Function

Algorithm 1: Rotation to Left and Right

```

/* RotL :  $\{0, 1\}^{32} \times \{0, 1\}^{32} \rightarrow \{0, 1\}^{32}$  */
1 Function RotL(value, shift):
2   | return (value  $\ll$  shift) | (value  $\gg$  (32 – shift));
3 end

/* RotR :  $\{0, 1\}^{32} \times \{0, 1\}^{32} \rightarrow \{0, 1\}^{32}$  */
4 Function RotR(value, shift):
5   | return (value  $\gg$  shift) | (value  $\ll$  (32 – shift));
6 end

```

1.3.3 Encryption Key Schedule of LEA-128

Algorithm 2: Encryption Key Schedule (LEA-128)

Input: User-key $UK = UK[0] \parallel UK[1] \parallel UK[2] \parallel UK[3]$ ($UK[i] \in \{0, 1\}^{32}$)

Output: Encryption Round-keys $\{RK_i^{\text{enc}}\}_{i=0}^{23}$ ($RK_i^{\text{enc}} \in \{0, 1\}^{192}$)

/* $UK \in \{0, 1\}^{128}$ is 16-byte and $\{RK_i^{\text{enc}}\}_{i=0}^{23} \in \{0, 1\}^{4608}$ is 576-byte */

```

1 for  $i = 0$  to 3 do
2   |  $T[i] = UK[i]$  //  $T = T[0] \parallel \dots \parallel T[3] \in \{0, 1\}^{128=32*4}$ 
3 end
4 for  $i = 0$  to 23 do
5   |  $T[0] \leftarrow \text{RotL}(T[0] \boxplus \text{RotL}(\delta[i \bmod 4], i + 0), 1)$  //  $T[i] \in \{0, 1\}^{32}$ 
6   |  $T[1] \leftarrow \text{RotL}(T[1] \boxplus \text{RotL}(\delta[i \bmod 4], i + 1), 3)$ 
7   |  $T[2] \leftarrow \text{RotL}(T[2] \boxplus \text{RotL}(\delta[i \bmod 4], i + 2), 6)$ 
8   |  $T[3] \leftarrow \text{RotL}(T[3] \boxplus \text{RotL}(\delta[i \bmod 4], i + 3), 11)$ 
9   |  $RK_i^{\text{enc}} \leftarrow T[0] \parallel T[1] \parallel T[2] \parallel T[1] \parallel T[3] \parallel T[1]$  //  $RK_i^{\text{enc}} \in \{0, 1\}^{196=32*6}$ 
10 end
11 return  $\{RK_i^{\text{enc}}\}_{i=0}^{23}$ 

```

1.3.4 Decryption Key Schedule of LEA-128

Algorithm 3: Decryption Key Schedule (LEA-128)

Input: User-key $UK = UK[0] \parallel UK[1] \parallel UK[2] \parallel UK[3]$ ($UK[i] \in \{0, 1\}^{32}$)

Output: Decryption Round-keys $\{RK_i^{\text{dec}}\}_{i=0}^{23}$ ($RK_i^{\text{dec}} \in \{0, 1\}^{192}$)

/ $UK \in \{0, 1\}^{128}$ is 16-byte and $\{RK_i^{\text{dec}}\}_{i=0}^{23} \in \{0, 1\}^{4608}$ is 576-byte */*

```

1 for  $i = 0$  to 3 do
2    $T[i] = UK[i]$                                 //  $T = T[0] \parallel \dots \parallel T[3] \in \{0, 1\}^{128=32*4}$ 
3 end
4 for  $i = 0$  to 23 do
5    $T[0] \leftarrow \text{RotL}(T[0] \boxplus \text{RotL}(\delta[i \bmod 4], i + 0), 1)$                                 //  $T[i] \in \{0, 1\}^{32}$ 
6    $T[1] \leftarrow \text{RotL}(T[1] \boxplus \text{RotL}(\delta[i \bmod 4], i + 1), 3)$ 
7    $T[2] \leftarrow \text{RotL}(T[2] \boxplus \text{RotL}(\delta[i \bmod 4], i + 2), 6)$ 
8    $T[3] \leftarrow \text{RotL}(T[3] \boxplus \text{RotL}(\delta[i \bmod 4], i + 3), 11)$ 
9    $RK_{23-i}^{\text{dec}} \leftarrow T[0] \parallel T[1] \parallel T[2] \parallel T[1] \parallel T[3] \parallel T[1]$                                 //  $RK_i^{\text{dec}} \in \{0, 1\}^{196=32*6}$ 
10 end
11 return  $\{RK_i^{\text{dec}}\}_{i=0}^{23}$ 

```

1.4 Encryption of LEA-128

Algorithm 4: Encryption of LEA-128

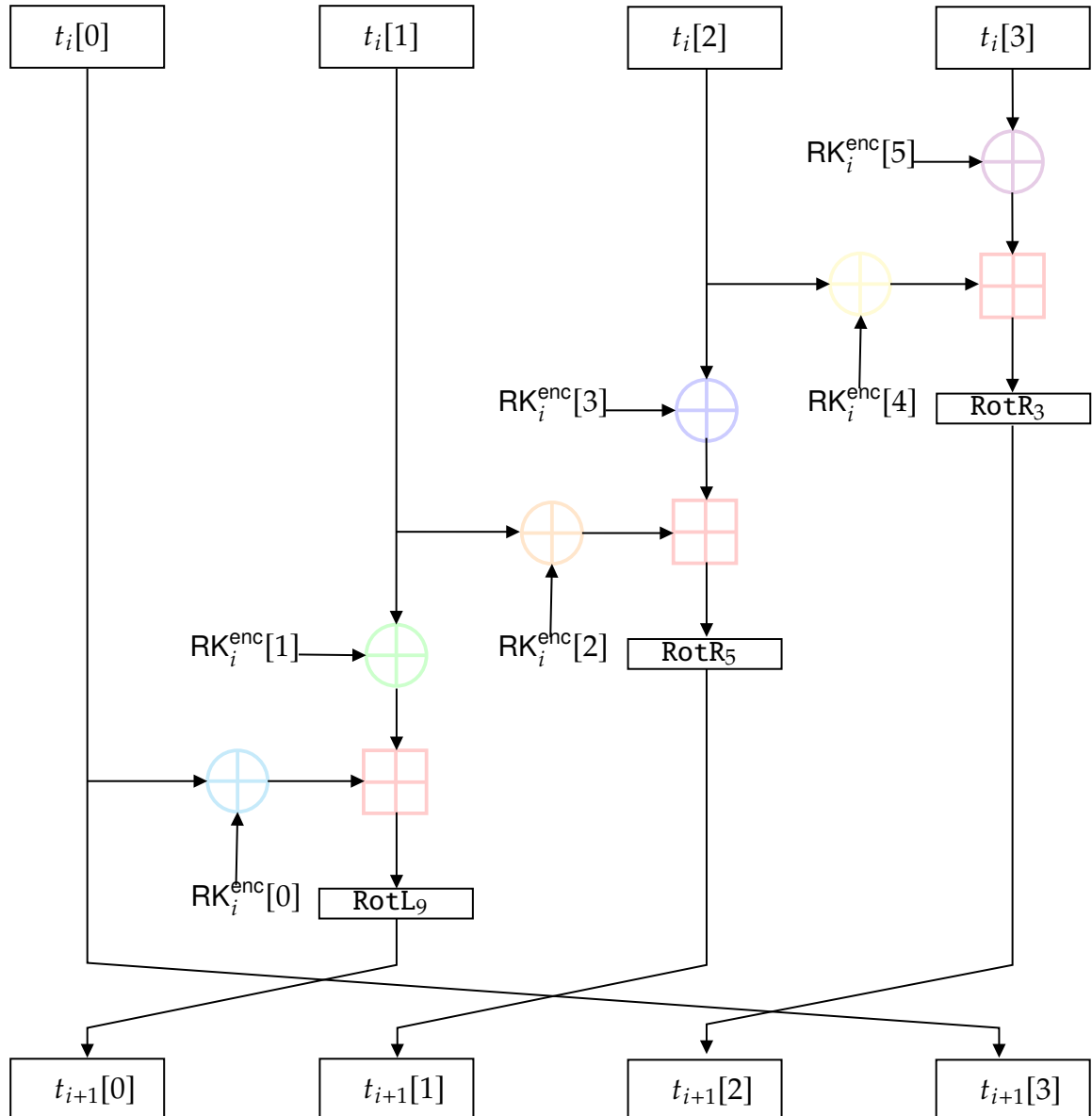
Input: block $\text{src} = \text{src}[0] \parallel \text{src}[1] \parallel \text{src}[2] \parallel \text{src}[3] \in \{0, 1\}^{128=32 \times 4}$ and $\{\text{RK}_i^{\text{enc}}\}_{i=0}^{N_r-1=23}$

Output: block $\text{dst} = \text{dsc}[0] \parallel \text{dsc}[1] \parallel \text{dsc}[2] \parallel \text{dsc}[3] \in \{0, 1\}^{128=32 \times 4}$

```

1  $t_0 = t[0] \parallel t[1] \parallel t[2] \parallel t[3] \leftarrow \text{src}$ 
2 for  $i = 0$  to 23 do
3    $\text{tmp} \leftarrow t[0]$ 
4    $t_{i+1}[0] \leftarrow \text{RotL}(t_i[0] \oplus \text{RK}_i^{\text{enc}}[0] \boxplus (t_i[1] \oplus \text{RK}_i^{\text{enc}}[1]), 9)$ 
5    $t_{i+1}[1] \leftarrow \text{RotR}(t_i[1] \oplus \text{RK}_i^{\text{enc}}[2] \boxplus (t_i[2] \oplus \text{RK}_i^{\text{enc}}[3]), 5)$ 
6    $t_{i+1}[2] \leftarrow \text{RotR}(t_i[2] \oplus \text{RK}_i^{\text{enc}}[4] \boxplus (t_i[3] \oplus \text{RK}_i^{\text{enc}}[5]), 3)$ 
7    $t_{i+1}[3] \leftarrow \text{tmp}$ 
8 end
9 return  $\text{dst} \leftarrow t_{N_r}$ 

```



1.5 Decryption of LEA-128

Algorithm 5: Encryption of LEA-128

Input: block $\text{src} = \text{src}[0] \parallel \text{src}[1] \parallel \text{src}[2] \parallel \text{src}[3] \in \{0, 1\}^{128=32*4}$ and $\{\text{RK}_i^{\text{enc}}\}_{i=0}^{N_r-1=23}$
Output: block $\text{dst} = \text{dst}[0] \parallel \text{dst}[1] \parallel \text{dst}[2] \parallel \text{dst}[3] \in \{0, 1\}^{128=32*4}$

```

1  $t_0 = t[0] \parallel t[1] \parallel t[2] \parallel t[3] \leftarrow \text{src}$ 
2 for  $i = 0$  to  $23$  do
3    $\text{tmp} \leftarrow t[0]$ 
4    $t_{i+1}[0] \leftarrow \text{RotL}(t_i[0] \oplus \text{RK}_i^{\text{enc}}[0] \boxplus (t_i[1] \oplus \text{RK}_i^{\text{enc}}[1]), 9)$ 
5    $t_{i+1}[1] \leftarrow \text{RotR}(t_i[1] \oplus \text{RK}_i^{\text{enc}}[2] \boxplus (t_i[2] \oplus \text{RK}_i^{\text{enc}}[3]), 5)$ 
6    $t_{i+1}[2] \leftarrow \text{RotR}(t_i[2] \oplus \text{RK}_i^{\text{enc}}[4] \boxplus (t_i[3] \oplus \text{RK}_i^{\text{enc}}[5]), 3)$ 
7    $t_{i+1}[3] \leftarrow \text{tmp}$ 
8 end
9 return  $\text{dst} \leftarrow t_{N_r}$ 

```

Algorithm 6: Decryption of LEA-128

Input: block $\text{src} \in \{0, 1\}^{128=8*16}$, decryption round-keys $\{\text{RK}_i^{\text{dec}}\}_{i=0}^{N_r-1=23}$
Output: block $\text{dst} \in \{0, 1\}^{128=8*16}$

```

1  $t_0 \leftarrow \text{src}$ 
2 for  $i = 0$  to  $N_r - 1$  do
3    $t_{i+1}[0] \leftarrow t_i[3]$ 
4    $t_{i+1}[1] \leftarrow (\text{RotR}(t_i[0], 9) \boxminus (t_{i+1}[0] \oplus \text{RK}_i^{\text{dec}}[0])) \oplus \text{RK}_i^{\text{dec}}[1]$ 
5    $t_{i+1}[2] \leftarrow (\text{RotL}(t_i[1], 5) \boxminus (t_{i+1}[1] \oplus \text{RK}_i^{\text{dec}}[2])) \oplus \text{RK}_i^{\text{dec}}[3]$ 
6    $t_{i+1}[3] \leftarrow (\text{RotL}(t_i[2], 3) \boxminus (t_{i+1}[2] \oplus \text{RK}_i^{\text{dec}}[4])) \oplus \text{RK}_i^{\text{dec}}[5]$ 
7 end
8 return  $\text{dst} \leftarrow t_{N_r}$ 

```

Appendix A

Additional Data A

A.1 Substitution-BOX