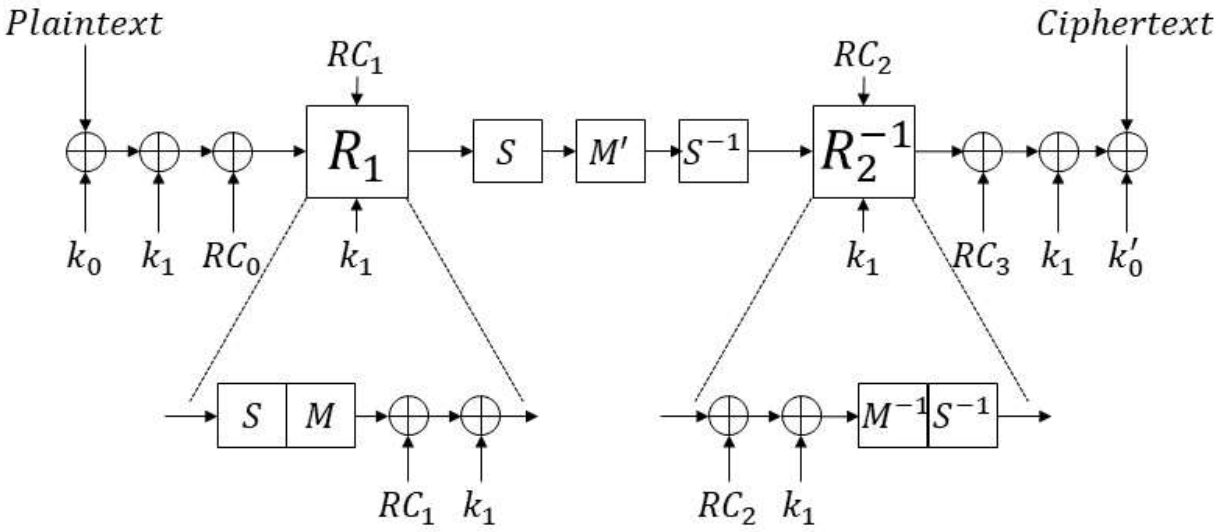
2021 암호분석경진대회

2번 문제 : 블록 암호

주어진 암호문 데이터 파일(ct.dat)은 평문 데이터 파일(pt.dat)을 4라운드로 축소된 PRINCE-64/128을 통해 암호화한 파일이다. 주어진 데이터 파일들을 활용하여 암호화에 사용된 128-bit 비밀키를 복구하시오. (단, 키는 알파벳 소문자로만 구성되어 있음. 또한, 각 데이터 파일은 litte-endian 방식으로 저장되어 있고 암호문 데이터 파일(ct.dat)에 적용된 운영모드는 추가적인 초기화 벡터를 사용하지 않고 블록 단위로 암호화를 수행하는 ECB 모드가 사용됨.)

※데이터 파일 다운로드
- 평문 데이터 파일: pt.dat
- 암호문 데이터 파일: ct.dat



(그림 1) 4-round PRINE-64/128 동작 과정

(그림 1)은 4-round PRINCE-64/128의 암호화 과정을 나타낸 그림이며 $\mathrm{PRINCE}_{core}(R_1)$은 라운드키 XOR, 라운드 상수 XOR, S-box 연산, 행렬곱 연산으로 구성된 라운드 함수를 말한다. 또한, k_0, k_0', k_1는 비밀키로부터 확장된 라운드키로 Key Expansion 함수를 통해 생성된다. (그림 1)의 세부 동작 과정 및 연산의 설명은 다음과 같다.

- 화이트닝 과정: (k_0-add.)
- 1 라운드: (k_1-add.) - (RC_0-add.)
- 2 라운드: R_1; (S-Layer) - (M-Layer) - (RC_1-add.) - (k_1-add.)
- Middle Layer: (S-Layer) - (M'-Layer) - (Inverse S-Layer)
- 3 라운드: R_2^{-1}; (RC_2-add.) - (k_1-add.) - (Inverse M-Layer) - (Inverse S-Layer)
- 4 라운드: (RC_3-add.) - (k_1-add.)
- 화이트닝 과정: (k_0'-add.)

또한, 암호화 동작 과정의 각 연산 및 라운드키 생성 과정 설명은 다음과 같다.

<PRINC-64/128 **암호화 설명**>

입력 평문은 다음과 같이 4-bit 니블로 분할되어 상태배열로 표현된다.

$$Plaintext = (p_0, p_1, ..., p_{14}, p_{15})$$

p_0	p_1	p_2	p_3
p_4	p_5	p_6	p_7
p_8	p_9	p_{10}	p_{11}
p_{12}	p_{13}	p_{14}	p_{15}

- k_1-add.($\oplus k_i$) : 상태배열에 대한 64-bit 라운드키 XOR 연산

- S-Layer(S) : 상태배열의 각 니블에 대한 치환 연산 (4-bit S-box 사용)

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S[x]$	B	F	3	2	A	C	9	1	6	7	8	0	E	5	D	4

- Inverse S-Layer(S^{-1}) : S-Layer의 역연산

- M/M'-Layer (M/M') : M'-Layer는 다음과 같이 정의된 64×64 행렬 M' 곱 연산

$$M_0 = \begin{pmatrix} 0000 \\ 0100 \\ 0010 \\ 0001 \end{pmatrix}, M_1 = \begin{pmatrix} 1000 \\ 0000 \\ 0010 \\ 0001 \end{pmatrix}, M_2 = \begin{pmatrix} 1000 \\ 0100 \\ 0000 \\ 0001 \end{pmatrix}, M_3 = \begin{pmatrix} 1000 \\ 0100 \\ 0010 \\ 0000 \end{pmatrix},$$

$$\hat{M}_0 = \begin{pmatrix} M_0 M_1 M_2 M_3 \\ M_1 M_2 M_3 M_0 \\ M_2 M_3 M_0 M_1 \\ M_3 M_0 M_1 M_2 \end{pmatrix}, \hat{M}_1 = \begin{pmatrix} M_1 M_2 M_3 M_0 \\ M_2 M_3 M_0 M_1 \\ M_3 M_0 M_1 M_2 \\ M_0 M_1 M_2 M_3 \end{pmatrix}$$

$$M' = (\hat{M}_0 \bullet \hat{M}_1 \bullet \hat{M}_1 \bullet \hat{M}_0)$$

M-Layer는 $SR \bullet M'$ 연산으로 정의된다.

여기서 SR은 상태배열의 각 열에 대한 순환이동 연산으로 다음과 같이 동작한다.

0	1	2	3
4	5	6	7
8	9	A	B
C	D	E	F

→

0	5	A	F
4	9	E	3
8	D	2	7
C	1	6	B

- Inverse M-Layer(M^{-1}) : M-Layer의 역연산($M' \bullet SR^{-1}$)

- RC_i-add.($\oplus RC_i$) : 상태배열에 대한 64-bit 라운드 상수 XOR 연산

RC_0	0000000000000000
RC_1	13198a2e03707344
RC_2	a4093822299f31d0
RC_3	082efa98ec4e6c89

<PRINCE-64/128 라운드키 생성 설명>

입력된 128-bit 비밀키는 64-bit 씩 분할($k = k_0 \| k_1$) 된 후 Key Expansion 함수를 통해 k_0'을 생성하여 192-bit 로 확장된다.

- **Key Expansion** : PRINCE에서 사용하는 라운드키 생성 함수로 다음과 같이 정의된다.

$$(K_0 \| K_1) \rightarrow (k_0 \| k_0' \| k_1) := (k_0 \| (k_0 \ggg 1) \oplus (k_0 \gg 63) \| k_1)$$

<4-Round PRINCE-64/128 참조 구현 값>

Testvector 1:
평문: 0000000000000000
암호문: E35168F91283502C
키: (k_0: 0000000000000000 || k_1: 0000000000000000)

Testvector 2:
평문: FFFFFFFFFFFFFFFF
암호문: 96775187FC6A9943
키: (k_0: 0000000000000000 || k_1: 0000000000000000)

Testvector 3:
평문: 0000000000000000
암호문: 6988AE78039566BD
키: (k_0: FFFFFFFFFFFFFFFF || k_1: 0000000000000000)

Testvector 4:
평문: 0000000000000000
암호문: C611A0EC10C574E4
키: (k_0: 0000000000000000 || k_1: FFFFFFFFFFFFFFFF)

Testvector 5:
평문: 0123456789ABCDEF
암호문: 2579F2F660306F5E
키: (k_0: 0000000000000000 || k_1: FEDCBA9876543210)

Testvector 6:
평문: 0123456789ABCDEF
암호문: E31CF8A9AE6A50C7
키: (k_0: FFFFFFFFFFFFFFFF || k_1: FEDCBA9876543210)

메모리 적재 상태 예시 (Little-endian: Low address <···> High address)

평문								k_0								k_0'								k_1								암호문							
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	2C	50	83	12	F9	68	51	E3
FF	FF	FF	FF	FF	FF	FF	FF	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	43	99	6A	FC	87	51	77	96
00	00	00	00	00	00	00	00	FF	FF	FF	FF	FF	FF	FF	FF	FE	FF	FF	FF	FF	FF	FF	FF	00	00	00	00	00	00	00	00	BD	66	95	03	78	AE	88	69
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	FF	FF	FF	FF	FF	FF	FF	FF	E4	74	C5	10	EC	A0	11	C6
EF	CD	AB	89	67	45	23	01	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	10	32	54	76	98	BA	DC	FE	5E	6F	30	60	F6	F2	79	25
EF	CD	AB	89	67	45	23	01	FF	FF	FF	FF	FF	FF	FF	FF	FE	FF	FF	FF	FF	FF	FF	FF	10	32	54	76	98	BA	DC	FE	C7	50	6A	AE	A9	F8	1C	E3