Side Channel Analysis Contest 2022			
3. LEA-128			
학교	국민대학교	이름	이현호

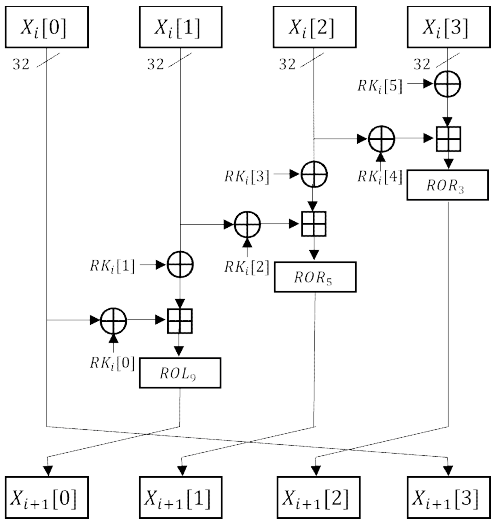
1. 블록 암호 LEA-128의 특징

본 라이트업에서 사용할 기호와 그에 대한 정의는 다음 [표 1] 과 같다.

기호	정의
$X_i[j]$	i 번째 라운드 X_i의 j번째 32 비트열 $(1 \le i \le 24, 0 \le j < 4)$
$\oplus$	eXclusive OR 연산
$\boxplus$	modulo 2^{32} 덧셈
$\boxminus$	modulo 2^{32} 뺄셈
$ROR_i(x)$	32비트 비트열 x의 i비트 오른쪽 로테이션 연산
$ROL_i(x)$	32비트 비트열 x의 i비트 왼쪽 로테이션 연산
$RK_i[j]$	i 번째 라운드 키의 j번째 32 비트열$(1 \le i \le 24, 0 \le j < 6)$
$\delta[i]$	키 스케줄 함수에서 사용되는 32비트 상수들 $(0 \le i \le 7)$

[표 1] 기호 및 정의

LEA-128의 i 번째 라운드 암호화 과정은 [그림 1] 과 같다. $(1 \le i < 24)$



[그림 1] LEA-128 암호화 과정

2. 1라운드 키 복구

LEA-128은 키 스케줄 함수로부터 생성된 라운드 키를 활용하여 암호화를 한다. LEA-128 에서 $\boxplus$ 연산만이 비선형 연산이다. 따라서 상관전력분석에서 가장 이상적인 중간값은 $\boxplus$ 연산 구간이다. 하지만 LEA-128의 한 라운드에서 $\boxplus$ 연산에 필요한 라운드 키는 2개이며 각 라운드 키 크기는 32-bit이다. 이는 분석 시간에 엄청난 영향을 주기 때문에 매우 비효율적인 방식이다.

그러므로 본 라이트업에서는 $\oplus$연산 구간에서 필요한 32-bit 라운드 키가 1개라는 점을 이용한다. 라운드 키를 1바이트 단위로 총 4개를 찾는다. 하지만 $\oplus$연산 구간을 중간값으로 하는 경우 같은 상관계수를 가지는 키 후보가 항상 2개씩 나오기 때문에 어느 키 후보가 옳은 키인지 확신하기 힘들다는 단점이 있다. 따라서 옳은 키를 확인하기 위해 키 후보들을 활용하여 $\boxplus$ 연산의 중간값으로 두고 다시 상관전력분석을 실행한다. 만약 잘못된 키라면 상관계수와 분석 결과 가장 큰 상관계수를 두 번째로 큰 상관계수로 나눈 값을 의미하는 비율이 낮게 나올 것이다.

위와 같은 전략을 이용하여 구한 1라운드 키는 [표 2]와 같다.

분석 위치	라운드 키
$RK_1[0]$	0x52cc3a5a
$RK_1[1]$	0xb249ffbf
$RK_1[2]$	0x4206ed9d
$RK_1[3]$	0xb249ffbf
$RK_1[4]$	0xc4a14a06
$RK_1[5]$	0xb249ffbf

[표 2] 1라운드 키

3. LEA-128 마스터 키 복구

1라운드 키를 알았기 때문에 키 스케줄 함수의 역함수를 이용하여 마스터 키를 복구한다.
32-bit 크기 입출력값 T[0], T[1], T[2], T[3] 에 대해 역함수는 [그림 2]와 같다.

$$T[0] \leftarrow ROR_1(T[0]) \boxminus ROL_0(\delta[0 \bmod 4])$$
$$T[1] \leftarrow ROR_3(T[1]) \boxminus ROL_1(\delta[1 \bmod 4])$$
$$T[2] \leftarrow ROR_6(T[2]) \boxminus ROL_2(\delta[2 \bmod 4])$$
$$T[3] \leftarrow ROR_{11}(T[3]) \boxminus ROL_3(\delta[3 \bmod 4])$$

[그림 2]

역함수를 이용하여 복구한 마스터 키는 [표 3]과 같다.

분석 바이트	마스터 키
1	0x52
2	0x33
3	0x76
4	0x65
5	0x40
6	0x6c
7	0x69
8	0x6e
9	0x47
10	0x74
11	0x48
12	0x65
13	0x4b
14	0x45
15	0x59
16	0x21

[표 3] 마스터 키

4. [표 3]의 마스터 키를 바탕으로 복구한 Flag 의 ASCII 코드는 다음과 같다.

R3ve@linGtHeKEY!