Side Channel Analysis Contest 2022			
4. ARIA-128			
학교	국민대학교	이름	이현호

1. 주어진 블록 암호 ARIA-128의 특징

ARIA의 Diffusion layer(확산 계층)에서는 16 × 16 involution 이진 행렬을 사용한다. 이 행렬을 A라고 이름 붙이겠다.

$$A = \begin{matrix} 0&0&0&1&1&0&1&0&1&1&0&0&0&1&1&0 \\ 0&0&1&0&0&1&0&1&1&1&0&0&1&0&0&1 \\ 0&1&0&0&1&0&1&0&0&0&1&1&1&0&0&1 \\ 1&0&0&0&0&1&0&1&0&0&1&1&0&1&1&0 \\ 1&0&1&0&0&1&0&0&1&0&0&1&0&0&1&1 \\ 0&1&0&1&1&0&0&0&0&1&1&0&0&0&1&1 \\ 1&0&1&0&0&0&0&1&0&1&1&0&1&1&0&0 \\ 0&1&0&1&0&0&1&0&1&0&0&1&1&1&0&0 \\ 1&1&0&0&1&0&0&1&0&0&1&0&0&1&0&1 \\ 1&1&0&0&0&1&1&0&0&0&0&1&1&0&1&0 \\ 0&0&1&1&0&1&1&0&1&0&0&0&0&1&0&1 \\ 0&0&1&1&1&0&0&1&0&1&0&0&1&0&1&0 \\ 0&1&1&0&0&0&1&1&0&1&0&1&1&0&0&0 \\ 1&0&0&1&0&0&1&1&1&0&1&0&0&1&0&0 \\ 1&0&0&1&1&1&0&0&0&1&0&1&0&0&1&0 \\ 0&1&1&0&1&1&0&0&1&0&1&0&0&0&0&1 \end{matrix}$$

1라운드 키 $rk_i (0 \le i < 16)$는 rk_0 =0xCB, rk_1 = *, rk_2 =0x16, rk_3 =0xA7, rk_4 =0x91, rk_5 =0xAA, rk_6 =*, rk_7 =0x47, rk_8 =0x4D, rk_9 =0xA2, rk_{10} =0xD8, rk_{11} =*, rk_{12} =*, rk_{13} =0x2B, rk_{14} =0xC8, rk_{15} =0x83 (*는 제공되지 않음) 값으로 주어져 있다.

Diffusion layer에서 16개의 1바이트 입력값을 각각 $p_0, p_1, \cdots, p_{14}, p_{15}$ 그리고 16개의 1바이트 출력값을 각각 $s_0, s_1, \cdots, s_{14}, s_{15}$ 라고 하자. 확산 계층의 연산결과는 [수식 1]과 같다.

$$\begin{aligned}
s_0 &= p_3 \oplus p_4 \oplus p_6 \oplus p_8 \oplus p_9 \oplus p_{13} \oplus p_{14} \\
s_1 &= p_2 \oplus p_5 \oplus p_7 \oplus p_8 \oplus p_9 \oplus p_{12} \oplus p_{15} \\
s_2 &= p_1 \oplus p_4 \oplus p_6 \oplus p_{10} \oplus p_{11} \oplus p_{12} \oplus p_{15} \\
s_3 &= p_0 \oplus p_5 \oplus p_6 \oplus p_{10} \oplus p_{11} \oplus p_{13} \oplus p_{14} \\
s_4 &= p_0 \oplus p_2 \oplus p_5 \oplus p_8 \oplus p_{11} \oplus p_{14} \oplus p_{15} \\
s_5 &= p_1 \oplus p_3 \oplus p_4 \oplus p_9 \oplus p_{10} \oplus p_{14} \oplus p_{15} \\
s_6 &= p_0 \oplus p_2 \oplus p_7 \oplus p_9 \oplus p_{10} \oplus p_{12} \oplus p_{13} \\
s_7 &= p_1 \oplus p_3 \oplus p_6 \oplus p_8 \oplus p_{11} \oplus p_{12} \oplus p_{13} \\
s_8 &= p_0 \oplus p_1 \oplus p_4 \oplus p_7 \oplus p_{10} \oplus p_{13} \oplus p_{15} \\
s_9 &= p_0 \oplus p_1 \oplus p_5 \oplus p_6 \oplus p_{11} \oplus p_{12} \oplus p_{14} \\
s_{10} &= p_2 \oplus p_3 \oplus p_5 \oplus p_6 \oplus p_8 \oplus p_{13} \oplus p_{15} \\
s_{11} &= p_2 \oplus p_3 \oplus p_4 \oplus p_7 \oplus p_9 \oplus p_{12} \oplus p_{14} \\
s_{12} &= p_1 \oplus p_2 \oplus p_6 \oplus p_7 \oplus p_9 \oplus p_{11} \oplus p_{12} \\
s_{13} &= p_0 \oplus p_3 \oplus p_6 \oplus p_7 \oplus p_8 \oplus p_{10} \oplus p_{13} \\
s_{14} &= p_0 \oplus p_3 \oplus p_4 \oplus p_5 \oplus p_9 \oplus p_{11} \oplus p_{14} \\
s_{15} &= p_1 \oplus p_2 \oplus p_4 \oplus p_5 \oplus p_8 \oplus p_{10} \oplus p_{15}
\end{aligned}$$

[수식 1] 확산 계층의 연산결과

1라운드에서 $p_0, p_1, \cdots, p_{14}, p_{15}$값들은 AddRoundKey와 치환 계층(S-Box 사용구간)을 거치며 각각의 아래첨자 인덱스와 일치하는 $rk_i (0 \le i < 16)$ 정보를 가지고 있다.

2. 라운드 키 분석

1라운드에서 s_0, s_{10}, s_{13} 는 공통적으로 p_6 값에 대해 rk_6 =* 로 1바이트의 알 수 없는 정보를 포함한다. 마찬가지로 s_5, s_8, s_{15}는 p_1 값에 대해 rk_1 =*, s_3, s_4, s_{14}는 p_{11} 값에 대해 rk_{11} =*, s_1, s_6, s_{11}는 p_{12} 값에 대해 rk_{12} =* 로 1바이트의 알 수 없는 정보를 포함한다. $s_0, s_{10}, s_{13}, s_5, s_8, s_{15}, s_3, s_4, s_{14}, s_1, s_6, s_{11}$ 들은 2라운드 입력값이기도 하다. 파형 정보는 2라운드 Diffusion layer부터 주어져 있다. 하지만 Diffusion layer에서 행렬 A의 특징에 의해(행렬

A의 1의 값에 의해) 2라운드 치환 계층(S-Box 사용구간)의 1바이트 출력값이 그대로 노출 되기 때문에 이를 이용하여 1라운드 키와 2라운드 키(2바이트)를 한 번에 추측하는 중간값을 설정할 수 있다.
1라운드 키와 2라운드 키를 찾은 다음부터는 각 라운드의 치환 계층을 1바이트를 중간값으로 설정하고 라운드 키를 찾을 수 있다. 그렇게 찾은 각 라운드 키는 [표 1] 과 같다.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1라운드 키	0xCB	0xD3	0x16	0xA7	0x91	0xAA	0x4D	0x47	0x4D	0xA2	0xD8	0x76	0xCE	0x2B	0xC8	0x83
2라운드 키	0xBA	0xE3	0x64	0x1F	0x9E	0x99	0x65	0xFC	0x3D	0xED	0x67	0xEA	0x8D	0x51	0x5E	0xA7
3라운드 키	0x3F	0x16	0xF4	0xE9	0xDC	0x1D	0x98	0x32	0xE0	0xE3	0x64	0x7A	0x13	0xE1	0x18	0xFE
4라운드 키	0x10	0xA0	0xDB	0xA0	0x09	0xEE	0x84	0xDD	0xD0	0xCD	0x60	0x61	0x33	0x22	0x62	0x66
5라운드 키	0x25	0x27	0x59	0x81	0x2F	0x6C	0x7C	0xCA	0x81	0x97	0x92	0x33	0x08	0x8A	0x28	0xCD

[표 1] 1~5 라운드 키

3. ARIA-128 마스터 키 복구

라운드 키 생성에서 쓰이는 4개의 128비트를 $W_0, W_1, W_2, W_3 (W_0 : Master Key, W_1 : W_0$로 부터 유도되는 값) 라고 하자. 그렇다면 1라운드 키를 $W_0 \oplus (W_1 \gg 19)$, 5라운드 키를 $W_0 \oplus (W_1 \gg 31)$로 쓸 수 있다. 1라운드 키와 5라운드 키를 Xor연산을 하고 이 값의 각 비트를 왼쪽으로 19비트 순환이동 시켜주면 $W_1 \oplus (W_1 \gg 12)$의 결과값을 얻을 수 있다. 1라운드 키와 5라운드 키는 "2. 라운드 키 분석" 에서 구했기 때문에 W_1의 마지막 전 비트를 0 혹은 1로 추측하면(2비트 추측) W_1의 값을 구할 수 있다.
구한 W_1 값의 각 비트를 오른쪽으로 19비트 순환이동 시켜준 후 1라운드 키와 Xor 연산한 결과는 [수식 2] 와 같다.

$$W_0 \oplus (W_1 \gg 19) \oplus (W_1 \gg 19) = W_0$$

[수식 2] 마스터 키 복구 수식

마스터 키 W_0는 [표 2]와 같다.

바이트 위치	마스터 키
1	0x4b
2	0x6f
3	0x52
4	0x45
5	0x61
6	0x43
7	0x72
8	0x59
9	0x70
10	0x74
11	0x4f
12	0x5a
13	0x7a
14	0x41
15	0x6e
16	0x67

[표 2] ARIA-128 마스터 키

4. 위의 결과를 바탕으로 복구한 Flag 의 ASCII 코드는 다음과 같다.

KoREaCrYptOZzAng